Università degli Studi di Trieste

Department of Physics

Bachelor Thesis

Morphological Classification of Galaxies by Deep Learning Techniques

Candidate
Sebastiano Zagatti

Supervisor
Prof. Mauro Messerotti
Co-Supervisor
Dott. Daniele Tavagnacco

ACADEMIC YEAR 2018-2019

A mio papà Renato, per avermi insegnato a credere sempre nelle mie scelte e a non abbandonare mai i miei sogni.
A mia mamma Nicoletta, per essere la migliore madre che un uomo possa desiderare di avere, per avermi sostenuto incondizionatamente lungo questo percorso e per tutto l'amore che ha saputo trasmettermi.

Abstract

English:

The volume of astronomical data, and in particular galaxy images, has been progressively increasing in the last decade. Hence, the implementation of artificial neural networks and machine learning techniques have been considered an effective approach to cope with the need of fast and automatic information processing in the field of image analysis. The goal of this thesis is to test the applicability limits of a customized MATLAB Convolutional Neural Network for Deep Learning, in order to automatically classify the morphology of elliptical, spiral, barred spiral and edge-on galaxies in pre-processed digital image datasets. To cope with the high computational demand, a GPU accelerated computer has been used. In the first part of this work, we provide an overview on the morphological classification of galaxies, and an introduction to machine learning and deep learning, with particular emphasis on the analysis of the typical structure of a convolutional neural network. To characterize the computing platform, technical details about the GPU accelerated computer are provided. The criteria of image selection are described and different options of image pre-processing are discussed. Furthermore, various results of the convolutional network training processes are considered and the relevant diagnostic graphs are discussed. At last, based on the promising outcomes obtained for the selected datasets, training process improvements and future work perspectives are elaborated.

Abstract

Italian:

Nell'ultimo decennio il volume di dati di tipo astronomico, in particolare immagini di galassie, è andato crescendo progressivamente. Di conseguenza, l'implementazione delle reti neurali artificiali e delle tecniche di deep learning nel campo del trattamento delle immagini, è stata considerata come un approccio efficace per fronteggiare la necessità di analizzare queste informazioni in maniera veloce e automatica . L'obiettivo di questa tesi è testare i limiti di applicabilità di una rete neurale convoluzionale costruita *ad hoc* e sviluppata in MATLAB per il deep learning, al fine di ottenere una classificazione morfologica di galassie ellittiche, spirali, spirali barrate ed edge-on utilizzando dataset di immagini pre-trattate. Per affrontare l'elevato sforzo computazionale si è utilizzato un computer accelerato a GPU. Nella prima parte di questo lavoro, faremo un excursus teorico sulla morfologia delle galassie e introdurremo il machine learning e il deep learning, con particolare attenzione all'analisi della tipica struttura di una rete neurale convoluzionale. Per caratterizzare la piattaforma di calcolo verranno fornite le caratteristiche tecniche del computer accelerato a GPU. Verranno descritti criteri di selezione delle immagini e verranno proposte diverse opzioni di trattamento delle stesse. Inoltre verranno considerati i diversi risultati ottenuti dai processi di training e verranno discussi i grafici di diagnostica. Infine, sulla base dei promettenti risultati ottenuti, verranno esposte alcune idee per migliorare i risultati del processo di training e le prospettive future.

Contents

Introduction

In the last decade, the volume of astronomical images of galaxies has been progressively growing, mainly due to technological progress and the advent of wide area digital cameras on large aperture telescopes. The availability of these large data sets has introduced the need for new methods in astronomical image analysis.
An approach to the galaxy morphological classification for these big datasets consists in manual classification, carried on by amateurs and volunteers through a web site, such as e.g. the "Galaxy-Zoo Project". Although every volunteer is capable of classifying only a limited number of galaxies, the large amount of human observers causes this method to be very effective. On the other hand, the limits obviously arise when considering the need for a quick data analysis.
Therefore, it is evident that an automatic tool for galaxy morphological classification is needed to cope with the larger and larger amount of data modern astronomy is going to cope with.
In this framework, the goal of this thesis work is the implementation of Deep Learning techniques, in particular a Convolutional Neural Network, in order to automatically recognize and classify elliptical (E), spiral (S), barred spiral (SB) and edge-on galaxies.
The thesis is organized as follows.
The galaxy morphological classification system (Hubble Sequence scheme) is outlined in Chapter 1, that reports some of the important applications of galaxy morphology, i.e., formation and evolution of galaxies as well as information about the star population.
Machine Learning and Deep Learning, with particular emphasis on Convolutional Neural Networks, are introduced in Chapter 2 by highlighting their typical applications and structure, and the choices made for the implemented MATLAB script. Technical details about the GPU accelerated computer and the benefits of multi-GPU computing are also reported in this chapter.
The process and criteria used to create image datasets are described in Chapter 3, as well as the different image pre-processing options used to improve the training process.
The results of the different training processes and the relevant diagnostic graphs are discussed in Chapter 4.
Conclusions are drawn and future perspectives are elaborated in Chapter 5.

Chapter 1

Morphological Classification of Galaxies

In this chapter, the two main schemes for morphological classification of galaxies are described in detail, and an overview on some of the main physical issues relevant to this topic is presented.

1.1 The Hubble Sequence

[1]Galaxies are the fundamental building blocks of the universe and can present various structures and compositions. The first step to understand galaxies is to classify them according to their morphology. Although being subjective, this kind of classification provides a good framework to discuss their qualitative properties, which are indicative of the physical ones.
Through the history of astronomy, different classifications have been proposed, but all their main features refer to the first galaxy morphology classification made by Edwin Hubble in 1926: the Hubble Sequence (Figure 1.1). The different types of galaxies are ordered in a sequence from early to late types and three main types can be identified: elliptical, lenticular and spiral galaxies. Spiral galaxies are, for their part, divided into normal and barred spirals; in addition Hubble included another class: irregular galaxies.

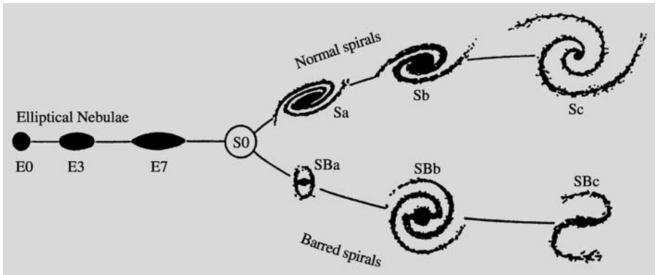


Figure 1.1: The Hubble Sequence in Hubble's 1936 version [2] [3].

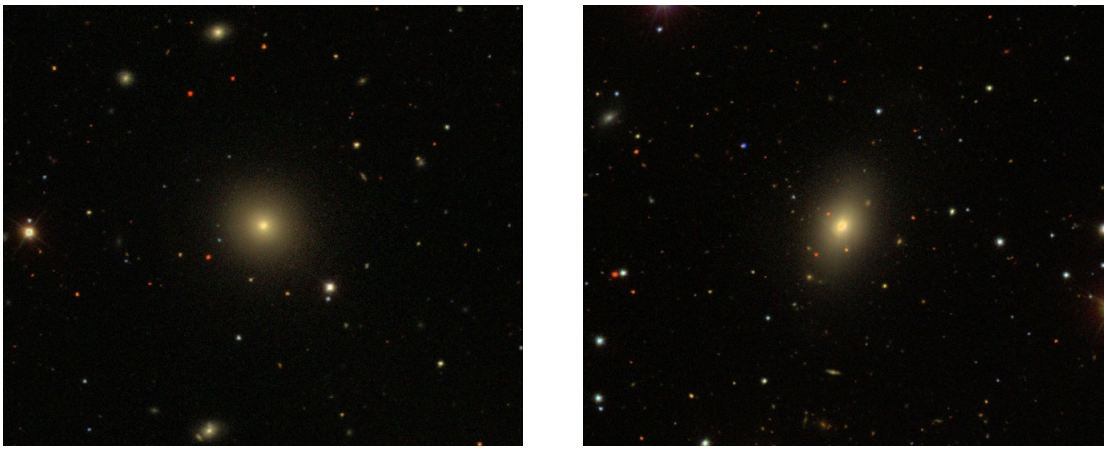
[1]Binney and Merrifield [1] and Karttunen et al. [2]

Elliptical Galaxies

The elliptical galaxies appear in the sky field images as elliptical concentrations of stars in which the density gradually falls off going outwards (Figure 1.2). Usually there are no signs of dark bands of dust or bright young stars, meaning that there is no interstellar matter.
The class of elliptical ones can be subdivided into eight sub-classes based on the shape: $E0$, $E1$, ..., $E7$. Considering a and b to be the major and minor axes of an elliptical galaxy, its type is defined to be E_n, with n given by:

$$n = 10\left(1 - \frac{b}{a}\right) \tag{1.1}$$

Generally speaking, in all galaxies the oldest stars are located as a round distribution, the inner parts are called "bulge" and the outer parts are referred to as "halo". It seems there is no significant difference between the bulge and the halo, thus the populations of old stars can be easily studied in ellipticals, which only contain this stellar component.
A later addition to the Hubble Sequence is the class of the "Giant Elliptical Galaxies", denoted as cD. These galaxies are usually found in the central part of clusters of galaxies and consist of a central part, that looks like a normal elliptical, which is surrounded by an extended fainter halo of stars.



(a) NGC 6109 (b) NGC 6021

Figure 1.2: Two examples of elliptical galaxies of different classes: (a) is an $E1$, (b) is an $E5$ SDSS [4]

Lenticular Galaxies

In the Hubble Sequence, the lenticular galaxies class occupies an intermediate position between the elliptical and the spiral types. These galaxies keep the usual elliptical stellar component, contain only little interstellar matter and show no sign of spiral structure, but they also contain a flat disc made of stars, which is a common feature of spiral galaxies (Figure 1.3).
Lenticular Galaxies are indicated as $S0$.

(a) NGC 4866 (b) NGC 7803

Figure 1.3: Two examples of lenticular galaxies [4].

Spiral Galaxies

The characteristic feature of spiral galaxies is the presence of a more or less well defined spiral pattern in the disc. Spiral galaxies usually consist of a central bulge, similar to an elliptical galaxy, and of a stellar disc, similar to the one featured by the lenticular ones; in addition to these, there is a thin disc of gas and other interstellar matter forming the spiral pattern, where young stars are being born (Figure 1.4).
As mentioned before, there are two sequences of spiral galaxies: normal spirals, listed as *Sa-Sb-Sc*, and barred spirals, listed as *SBa-SBb-SBc*.
These two sub-types actually occur with similar frequencies and their difference is not always evident.
In the barred spirals, the spiral pattern ends at a central bar, whereas in the normal spirals the pattern may end at an inner ring or continue all the way down to the centre.

The classification of a galaxy within the spiral sequence is determined according to three criteria:

1. Bulge size (later types have a smaller bulge);
2. Spiral arms size (later types have narrower spiral arms);
3. Spiral pattern (later types have a more open spiral pattern);

This superposition of criteria is highly unsatisfactory, but it turns out that there is sufficient correlation among these three parameters in order to build an unambiguous classification.

(a) NGC 7606

(b) NGC 3184

(c) NGC 6408

(d) NGC 0200

Figure 1.4: Four examples of spiral galaxies: (a) is an Sb, (a) is an SC, (c) is an SBa and (d) is an SBc [4].

Irregular Galaxies

Objects that lack symmetry or well defined spiral arms were classified as Irregular Galaxies and listed as Irr.

1.2 The de Vaucouleurs System

[2]Hubble's original scheme has been considered satisfactory for elliptical galaxies, but many astronomers argued that Hubble's classification of spirals was incomplete and considered his treatment of irregular galaxies as inadequate.
In particular, Gérard de Vaucouleurs proposed a more elaborate classification (Figure 1.5). He extended Hubble's turning fork by three additional classes for each sequence: Sd, Sm and Im.
The Sd class overlaps Hubble's Sc to some extent, but it also contains some more extreme objects that were classified as Irr galaxies before.
The Sm and Ir classes contain the remaining galaxies of Hubble's Irregular class, the "m" stands for "Magellanic", because the Large Magellanic Cloud was classified this way.
De Vaucouleurs also introduced a different notation for the unbarred spiral galaxies, listing them as SA, while the barred spirals were kept as SB. He also added a new sequence, that of the "weakly barred" or "mixed" galaxies, listed as SAB (Figure 1.6).

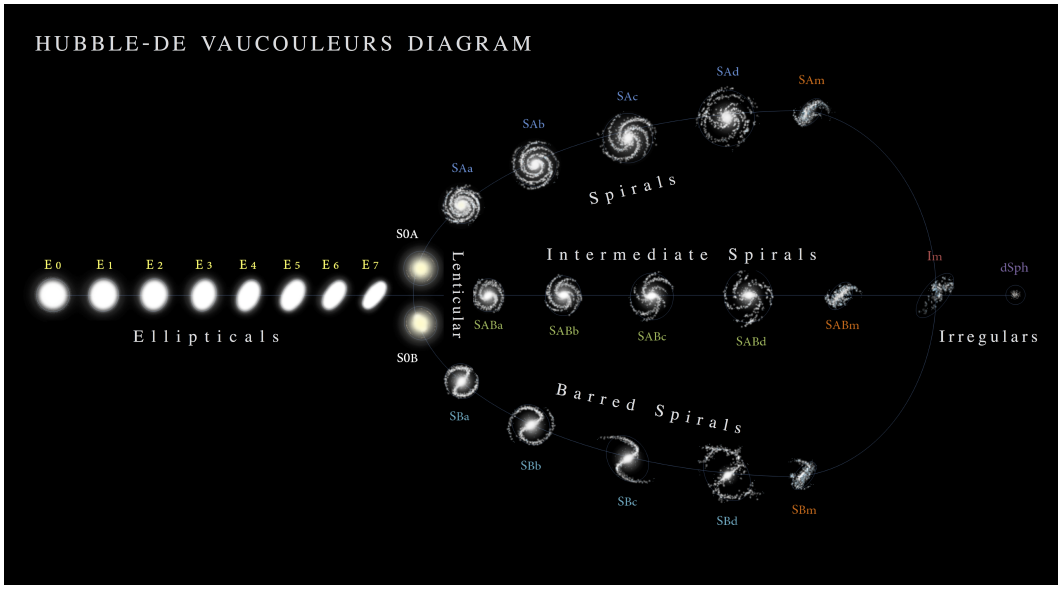


Figure 1.5: The de Vaucouleurs system [5].

[2]Binney and Merrifield [1]

(a) NGC 2805 (b) NGC 5204 (c) NGC 4068

(d) UGC 09057 (e) NGC 3664 (f) NGC 0014

Figure 1.6: Six examples of the classes added by de Vaucouleurs: (a) is an SAd, (b) is an SAm, (c) is an IAm, (d) is an SBd, (e) is an SBm, (f) is an IBm [4].

1.3 The Importance of Galaxy Morphology

1.3.1 Galactic Structures

Elliptical Galaxies and Bulges

[3]The surface brightness distribution in elliptical galaxies depends only on the distance from the centre and the orientation of the two axes. Being r the radius along the major axes, the surface brightness $I(r)$ is described by de Vaucouleurs' law:

$$\log \frac{I(r)}{I_e} = -3.33 \left[\left(\frac{r}{r_e} \right)^{1/4} - 1 \right] \tag{1.2}$$

where the constants are chosen so that half of the galaxy's total light is radiated within the radius r_e and the surface brightness at that radius is I_e. The parameters r_e and I_e are obtained by fitting (1.2) to observed profiles.
De Vaucouleurs' law is therefore a purely empirical relation, but it still gives a good representation of the observed light distribution, with some exceptions regarding peculiar outer regions of the ellipticals and the giant galaxies of type cD. Although the isophotes in elliptical galaxies are ellipses to a good approximation, their ellipticities and the orientation of their major axes may vary as a function of radius. Different galaxies differ widely in this aspect, indicating that the structure

[3]Karttunen et al. [2]

of ellipticals is not as simple as it might appear.
Equation (1.2) corresponds to a straight line and fits well for an E galaxy, but in the cD case the luminosity falls off more slowly in the outer regions. Comparison with Figure 1.8 shows that the brightness distribution of $S0$ galaxies behaves similarly to cD, leading to some erroneous classifications. [2]

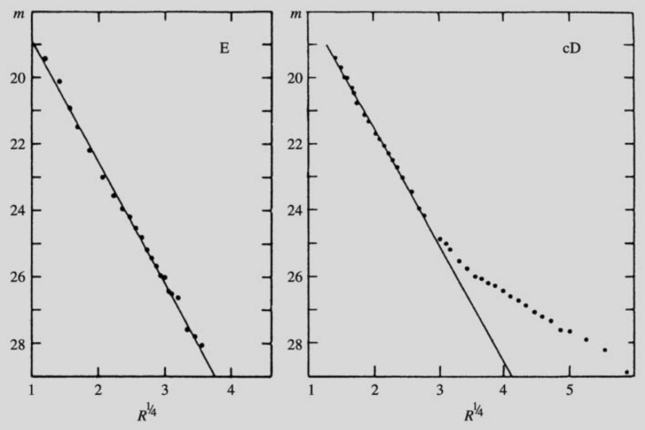


Figure 1.7: The distribution of surface brightness in E and cD galaxies. Ordinate: surface magnitude [mag/sq.arcsec]. Abscissa: radius $[\text{kpc}]^{1/4}$.

Disc

A bright and massive stellar disc is typical in $S0$ and spiral galaxies, which are therefore called disc galaxies. The distribution of surface brightness in the disc is described by:

$$I(r) = I_0 \exp^{-r/r_0} \tag{1.3}$$

Figure 1.8 shows how the observed radial brightness distribution can be decomposed into a sum of two components: a centrally dominant bulge and a disc contributing significantly at larger radii.

1.3.2 Stellar Ages and Element Abundances in Galaxies

[4]The most easily measured indicators of composition are the variations of colour indices inside galaxies and between different galaxies. Two regularities have been discovered: first, according to the colour-luminosity relation for elliptical and $S0$ galaxies, brighter galaxies are redder. Secondly, there is a colour-aperture effect, so that the central parts of the galaxies are redder.
Galactic spectra are composed of the spectra of all their stars added together, thus the colours depend both on the ages of the stars (young stars are bluer) and on the heavy element abundance Z (stars with larger Z are redder). Therefore, the interpretation of the observational results has to be based on detailed modelling of

[4]Karttunen et al. [2]

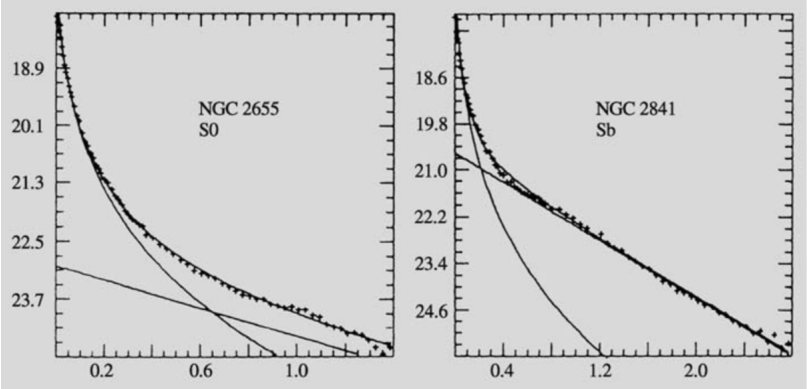


Figure 1.8: The distribution of surface brightness in $S0$ and Sb galaxies. Ordinate: surface magnitude [mag/sq.arc sec]. Abscissa: radius [arc sec]. The observed surface brightness has been decomposed into the sums of bulge and disc components. It is notable that the disc component is larger in type Sb. [2]

the stellar composition of galaxies or population synthesis.
Population synthesis of E galaxies show that practically all their stars were formed simultaneously about 1.5×10^9 years ago. Most of their light comes from red giants and most of their mass resides in lower main sequence stars of less than one solar mass. Since all stars have the same age, the colour of elliptical galaxies is directly related to their metallicities.
The stellar composition of disc galaxy bulges is generally similar to that of ellipticals. The element abundances in the gas in spirals can be studied by means of the emission lines from H II regions ionised by newly formed stars. The metallicity increases towards the centre, but the size of the variation changes in different galaxies and is not yet well understood.

1.3.3 Effects of Environment

[5]The wide range of galaxy morphologies raises the issue of what causes this diversity. One possible answer to this question can be found by studying the environments in which galaxies of different types are found.
It has been observed that galaxies in clusters are much more likely to be ellipticals or lenticular than those in field galaxies. This consideration suggests that environmental factors play an important role in determining the morphology of the galaxies.
Not all clusters are the same: in 1974 Oemler [6] found that in some clusters up to 40% of the galaxies are elliptical, while in others the proportion is 15%. The so-called elliptical fraction, $f(E)$, was found to correlate with the morphology of the cluster: clusters with large values of $f(E)$ tend to have a regular and symmetric appearance, a cD galaxy is often found at their centre, while clusters with a small value of $f(E)$ generally have an irregular shape.

[5]Binney and Merrifield [1]

Oemler also provided the first quantitative evidence that the balance of morphological types varies within individual clusters, providing the first example of a morphology-radius relation in clusters. Considering a line of sight through a centrally-concentrated cluster, the projected number density of galaxies decreases monotonically with the radius R. Omler actually discovered that the projected number density of spiral galaxies increases with R within the cluster, thus the fraction of spiral galaxies in these systems, $f(Sp)$, increases with the distance from the centre. These observations were found to be consistent with later discoveries of there being almost no spiral galaxies in the cores of regular clusters.
Similarly in 1977 Melnick and Sargent [7] compared the ratio of the number density of spiral galaxies to that of $S0$ galaxies, finding that the latters become increasingly dominant at small values of R. Studies at different wavelengths showed the same trends.
The spatial segregation of galaxy types in clusters should result in measurable differences between the kinematics of galaxies belonging to different morphological types. Indeed, spiral galaxies must lie far from the centre of clusters in order to follow more energetic orbits through the cluster.
The first large scale study of morphological segregation was made by Dressler et al. [8], where he concluded that the correlations involving the radius of a cluster are not fundamental ones and that galaxy type is dictated by the local density of galaxies. Thus, the fundamental relations are of the morphology-density type, correlating the local galaxy densities with $f(Sp)$ and $f(S0)$ respectively.
The works of Beers and Tonry [9] and Merrifield and Kent [10] found that the distribution of galaxies in clusters is such that N and R are closely correlated, so if morphology is correlated with N, it is obviously correlated with R too, and vice-versa.

Chapter 2

Machine Learning and Deep Learning

In this chapter, Machine Learning, Deep Learning and Convolutional Neural Network are considered, with particular emphasis on some general concepts.
A detailed description of the architecture of a Convolutional Neural Network is reported, followed by some details about the MATLAB script implemented in this thesis for the automatic classification of galaxies from digital images, a powerful mean to study their physical properties as pointed out in Chapter 1.

2.1 Artificial Neural Networks

[2]An Artificial Neural Network is a mathematical computing system that learns to perform tasks by considering examples without being programmed with task-specific rules. These results are obtained through a series of interconnected nodes, known as artificial neurons. Each neuron receives an input and then processes it producing an output signal that can be passed to other neurons connected with it. Typically, neurons are aggregated into layers and different layers can perform different transformations of the data; multiple connection patterns are possible between the layers. A typical neural network consists of an input layer, that receives the external data, at least one hidden layer, that processes the data, and an output layer, that produces the results (Figure 2.1).
Classification techniques are often based on artificial neural networks that predict discrete responses, classifying input data into pre-determined set of categories. Common examples are medical imaging and speech recognition.
Regression techniques, on the other hand, predict continuous responses such as outputs that can take any value within a certain range, for example stock pricing and acoustic signal processing.

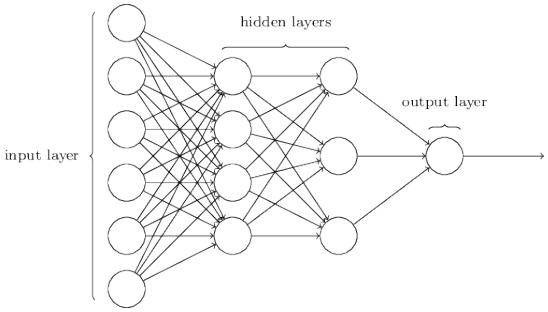


Figure 2.1: Graphical scheme of a basic neural network. [11]

2.2 Machine Learning

[1]Machine Learning (ML) is a subset of artificial intelligence that studies the algorithms and statistical models that computer systems use to perform a specific task without using specific instructions. Machine learning algorithms build a mathematical model based on sample data in order to make predictions or decisions without being explicitly programmed. It is usually used to solve complex problems involving a large amount of data with lots of variables, but no existing formula that describes the system. Some common applications of machine learning are:

- Systems that are too complex for handwritten rules, for example face and speech recognition;
- Situations where the rules of a task are constantly changing, as in fraud detection;
- Systems based on continuously changing data, like automated trading, energy demand forecasting and shopping trends prediction.

Machine Learning uses two different types of approaches: unsupervised and supervised learning.

2.2.1 Unsupervised Learning

[2]Unsupervised learning draws assumptions from datasets that don't have labeled responses associated with the input data. This technique is used to explore datasets and reduce their complexity by downsizing their dimensionality or number of features.
The most common form of unsupervised learning techniques is cluster analysis, which separates data into groups based on shared characteristics. Its possible applications are genes sequence analysis, market research and object detection.

[1]Mathworks [12]
[2]Mathworks [13], [14], [15]

2.2.2 Supervised Learning

[2]Supervised learning uses both a known input dataset, called training set, and labeled output data, to train a model to map inputs to outputs and then predict the response to any new set of input data. In order to do so, the network uses both classification and regression techniques.

2.3 Deep Learning

[3]Deep Learning (DL) is a machine learning method based on neural networks architectures, that learns features and tasks directly from data; the term "deep" usually refers to the large number of hidden layers in the neural network (Figure 2.2).

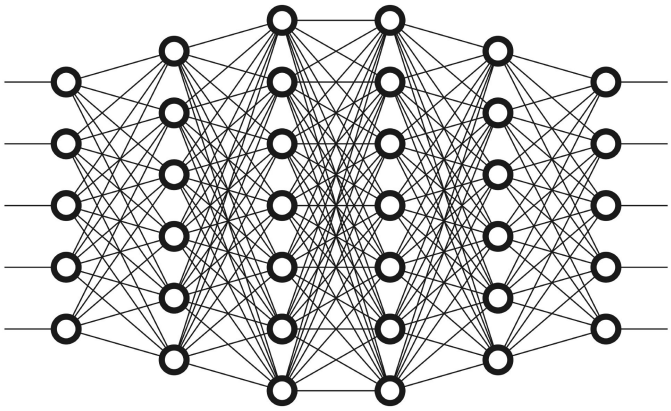


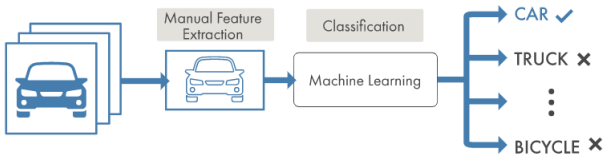
Figure 2.2: Graphical of a deep learning neural network. [19]

Deep Learning is often referred to as end-to-end learning because a set of labeled training data is used and needed by the algorithm, in order to learn the feature of each category that is going to be classified. Therefore the task is learned directly from the data.
That being said, deep learning could be defined as a Classification Supervised Machine Learning architecture.
This kind of technology has many applications in fields such as computer vision, audio and speech recognition, social network filtering, bioinformatics, medical image analysis and board games programs.

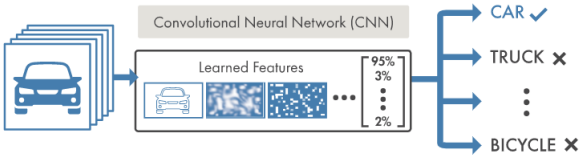
[3]Mathworks [16], [17], [18]

2.3.1 Differences between ML and DL

[3]Both ML and DL offer different ways to automatically classify data.
To have a computer to carry out a classification using a standard Machine Learning approach, it is necessary to manually select the relevant features of the considered data. After having selected them, the model references those features when analyzing and classifying new input data.
When solving a Machine Learning problem concerning image classification, a specific workflow is implemented (Figure 2.3(a)): firstly, a set of images is considered; secondly, relevant features of the images are selected (such as edges or corners); finally, a model that categorizes the objects in the images is created.
On the contrary, when considering Deep Learning workflow (Figure 2.3(b)), the manual step of extracting features is skipped, images are elaborated directly by the algorithm which then predicts and describes the images.
Another key advantage of Deep Learning networks is that they often continue to improve as the size of inserted data increases.



(a) Machine Learning



(b) Deep Learning

Figure 2.3: Graphical scheme of the differences between Machine Learning and Deep Learning. [17]

2.3.2 The Success and Outburst of Deep Learning

[4]Although being first theorized in the 1980s, the use of Deep Learning has been growing only in recent times, gaining popularity in all scientific fields. But why is Deep Learning becoming so popular over other Machine Learning techniques? And what are the reasons for this recent outburst?
The answer to the first question can be expressed in one word: accuracy.
Deep Learning achieves recognition accuracy at higher levels than ever before,

[4]Mathworks [16], [17], [18]

helping consumer electronics meet user expectations and being fundamental for safety critical applications such as driver-less cars. Recent advances have improved to the point where Deep Learning outperforms humans in some tasks, like objects classification in images. To answer the second question, two different factors have to be taken into account:

- Deep Learning requires large amounts of labeled data, which have become accessible only over the last few years, due to technological advancement;
- Deep Learning requires huge computational power to be achieved in a time-efficient way. The recent development of high-performance GPUs (Graphics Processing Units) with parallel architecture has boosted the running efficiency of the algorithms, drastically reducing training time.

2.4 Architecture of the Implemented DL Script

2.4.1 Convolutional Neural Networks

[5]A Convolutional Neural Network (CNN) is one of the most popular network architectures for Deep Learning. It is usually made up of several 2D layers that process and transform an input to produce an output. This particular architecture provides a well suited framework to process images and carry out automatic object recognition and classification (Figure 2.4).
Three of the most common layers for feature recognition and learning are respectively:

1. Convolution layer: it puts the input images through a set of convolutional filters, each of which activates certain features from the images;
2. Rectified Linear Unit (ReLU) layer: it fastens the training by mapping negative values to zero and maintaining positive values. It is also referred to as "Activation", because only the activated features are transmitted to the next layer;
3. Pooling layer: it simplifies the output by performing downsampling and reducing the number of parameters needed by the network to learn.

After learning the features, the architecture of a CNN is tailored to classification:

4. The next-to-last layer is a Fully Connected layer that outputs a vector whose dimension is the same as the number of classes that the network will be able to distinguish. This vector contains the probabilities for each class of any image being classified;
5. The final layer is a Softmax layer, that provides the classification output.

[5]Marchiori [15]; Mathworks [20], [21]

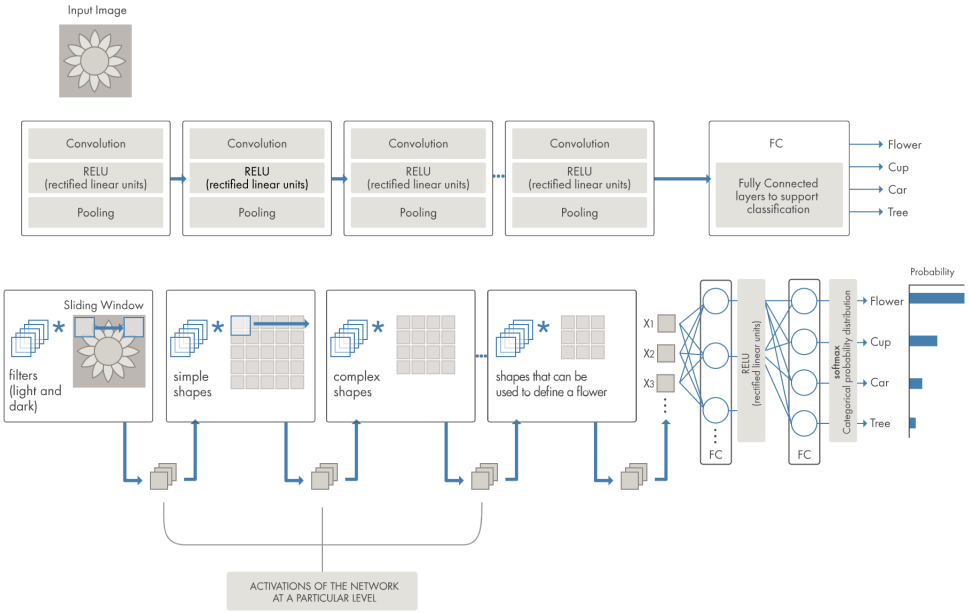


Figure 2.4: Sample scheme of a CNN used for object classification in images [17].

2.4.2 CNN Architecture

[6]In order to implement the described CNN, the MATLAB computing environment has been chosen. MATLAB provides a complete set of libraries for artificial neural network implementation and training, as well as many examples concerning Deep Learning techniques. Indeed, the final version of the implemented script used in this thesis was built on a pre-existing MATLAB template [22] regarding the classification of the MNIST database [24].
In the following section the typical structures of the MATLAB template will be presented:

1. **Image Dataset Loading**

   First of all, it is needed to specify the directory path storing the sample images. The images are organized into different labeled folders, each folder corresponding to a different class that the CNN will be required to classify. The script will then count the number of images in each folder and display the results in the Command Window, it will also determine the size of the images. Then, each label of the dataset will be randomly split into a training set and a validation set: the number of training images can be set by the programmer, whereas the number of validation images is given by the total number of images minus the number of training images.

[6]Marchiori [15]; Mathworks [22], [23]

2. **Network Architecture**

   The various types of layers used and their function are respectively:

   a. Image Input Layer

      The image input layer normalizes the input images and passes them to the following layers. It requires information about the size of the images, that has to be the same for all of them, and their format (grayscale or true color);

   b. Convolutional Layer

      A 2-D convolutional layer applies vertical and horizontal sliding filters to the input images. It is possible to define the height and the width of the filters, corresponding to the dimensions of kernel matrix used for filtering, as well as their number. In image processing, it is always difficult to define how to process the edge pixels of an image, a common way is the padding operation, which consists in adding virtual pixels with attribute zero externally to the edge pixels of the image. Padding is an option that is possible to select in the 2-d convolutional layer;

   c. Batch Normalization Layer

      A Batch Normalization layer normalizes each input channel independently, speeding up the training process;

   d. ReLU Layer

      A Rectified Linear Unit applies a threshold operation to the image by setting to zero any negative value. This layer is usually used in sequence with the Batch Normalization Layer;

   e. Max-Pooling Layer

      A max pooling layer performs down-sampling by dividing the input into rectangular pooling regions and computing the maximum of each region. This operation is very important as it allows to remove any redundant information and makes it possible to use many convolutional layers without increasing the computational effort;

   f. Fully Connected Layer

      The Fully Connected Layer usually follows the Max-Pooling Layer. All neurons of this layer connect with those of the previous one, making it possible to identify larger patterns. This layer is characterized by an output size parameter, that corresponds to the number of classes the CNN is going to be able to classify;

   g. Softmax Layer

      The Softmax Layer usually follows the Fully Connected Layer. It provides classification probabilities associated with each final class;

h. Classification Layer

The Classification Layer is the final layer of the neural network. It assigns each image to one of the final classes provided by the Fully Connected and Softmax layers.

3. **Training Options**

This section describes the possible options regarding the network's training process:

a. Training Algorithm MATLAB provides many different training algorithms. In this thesis a "Stochastic Gradient Descent with Momentum" (sdgm) was chosen. This iterative method can be used to minimize the error function by calculating the negative gradient of the loss function. The parameters are updated at each iteration using a subset of the training set (mini-batch), while the gradient is computed using the entire set.

b. Epochs and Mini-Batch Size

An epoch is a full pass of the algorithm over the entire training set. The number of epochs and the dimension of the mini-batch size can be specified by the programmer. The mini-batch size determines the number of iterations per epoch.

c. Validation Data and Validation Frequency

The Validation Data consists of a directory containing a subset of the original dataset, which is used to compute the total accuracy of the training process. This directory is created in the first pass of the script where the images have been split into two subsets. The `ValidationFrequency` parameter sets the number of times the network is validated per epoch.

d. Plot of the Training Process

This option opens a window containing the diagnostic plots of the training process. The first plot displays the progress of the training and the evolution of the accuracy percentage; the second plot displays the information loss relevant both to the training and to validation.

4. **Network Training and Final Accuracy**

In the end, the network is trained and the final accuracy of the trained network is computed. The accuracy corresponds to the fraction of image categories that the network is able to correctly classify from the Validation Dataset.

2.4.3 The Implemented Deep Learning Script

As mentioned in the previous sections, the final Deep Learning script implemented in this thesis was built on an existing template [22] provided by MATLAB. This template was built in order to classify the MNIST handwritten digits database [24] and consisted of 15 layers arranged in succession. The MNIST database is a commonly used, quite vast and really simple dataset of images (Figure 2.5), so that a CNN with many layers provides really good classification results, as depicted in Figure 2.6.

Figure 2.5: A few samples taken from the MNIST dataset.[25]

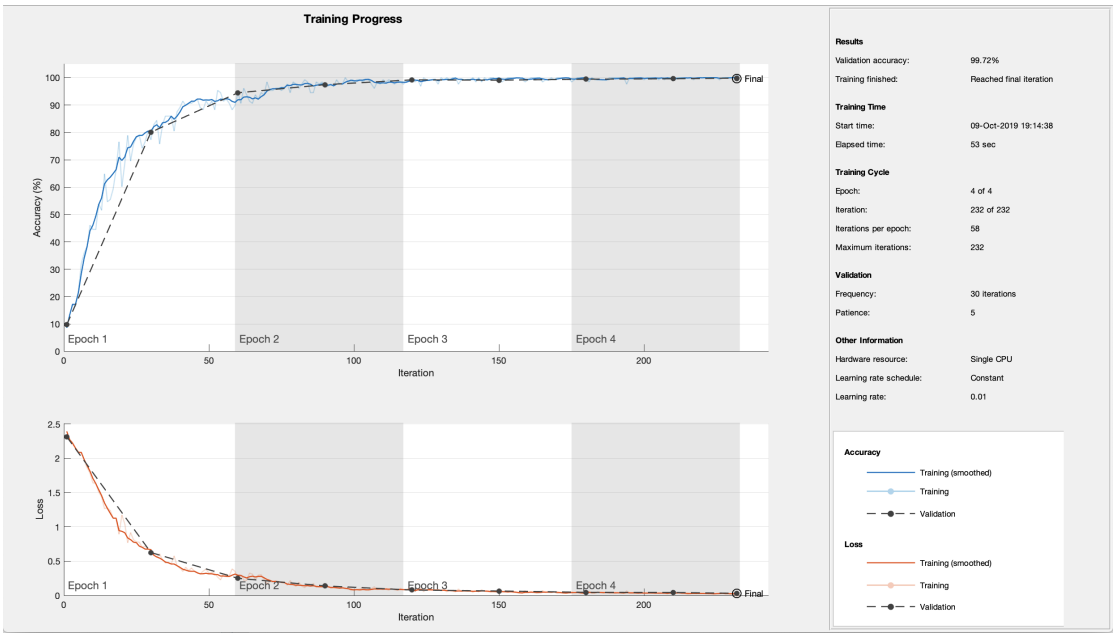


Figure 2.6: Training plots of the MATLAB template applied on the MNIST dataset; the accuracy is 99.72%.

The goal of this thesis is to classify the shapes of galaxies from sky field images which looks like an easy task, but it actually is a quite complex one, as will be elaborated in the following. Therefore, considering the simplicity of the MNIST dataset and the fact that it consists of over 10000 images, if compared with the complexity of galaxy images and the fact that the selected galaxy dataset contains 450 to 600 images (150 for each class), a tailoring of the script was needed.
The first adjustment made on the template script consisted of reducing the number of layers. The results that will be presented in Chapter 4 are relevant to networks consisting of 8 layers.
Although being one of the perks and the reason of deep learning's high efficiency, the great number of layers results in worst outcomes when coping with critical issues such as the complexity exhibited by galaxy images and the reduced dimension of the image dataset.
An additional Max-Pooling Layer was also added in the neural network architecture; the perks of this adjustment and the different classification results obtained with and without this addition are discussed in Chapter 4.1.
The second modification was changing the execution environment in the training options to 'multi-GPU', thus enabling the use of multiple GPUs and using a local parallel pool.
The third adjustment was introduced in order to cope with an heterogeneous distribution of the number of images selected for each class.
The script works with any number of images per class but, as previously mentioned in the first point of Section 2.4.2, only the dimension of the training set can be specified. Therefore, it is clear that an heterogeneous distribution of the number of images in the classes will cause the validation set to present a higher number of images for some classes with respect to other ones, causing a bias in the results of the validation process. Hence, the code was modified in order to randomly select 150 images for each class, to be used by the network in training and validation.
The last implemented modification was indeed another addition.
A further validation process was added in order to determine the behaviour and the accuracy of the already-trained network when facing an external set of data: 25 images are randomly selected from the images left out in the first selection and used as a second validation set.

2.5 The GPU Accelerated Computer

As stressed in Section 2.3.2, Deep Learning involves high computing demand. If a time-efficient training is needed, the computation can only be achieved by high-performance GPUs. During the simulations needed for this thesis, the HP Proliant SX40, a GPU accelerated computer, was used. This server is part of the computing infrastructure of the COSMOS project of the Italian Space Agency and running time was kindly granted by the INAF-Astronomical Observatory of Trieste.

2.5.1 Technical Specifications

The hardware architecture of the HP Proliant SX40 is based on the following components:

a. Two Intel Xeon Skylake Gold 6130 processors, 16 cores each and a basic working clock frequency of 2.10 GHz that can be boosted up to 3.70 GHz;

b. Four Tesla V100 Nvidia GPUs with a capacity of 16 GB, connected through the NVIDIA NVLink™[26]. These are the most advanced data center GPUs ever built to accelerate artificial intelligence, high performance computing and data science, delivering 47×higher inference performance than a common CPU server;

c. Twelve 16 GB DDR4 RAMs for a total of 192 Gb of RAM;

d. Two 10 Gigabit Ethernet Network Interface Cards;

e. Two 240 GB solid state disks (SSD) for data storage.

2.5.2 Application Containerization

The computer's workspace is structured in containers (Figure 2.7). Application Containerization is an OS-level virtualization method, used to deploy and run distributed applications without launching an entire virtual machine. Application containers consume fewer resources than a comparable deployment on virtual machines, because app containers share resources without a full operating system to underpin each app. Nevertheless, every container is isolated and processes in user space [27].

2.5.3 GPUs vs CPUs

A central processing unit (CPU) is the electronic circuitry within a computer that carries out the instructions of a computer program by performing the basic arithmetic, logic, controlling and input/output operations.
A graphics processing unit (GPU) is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images to be displayed on an external device.
GPUs appeared as a response to graphically intense applications that set a burden on the CPU and degraded computer performances, becoming a way to offload those tasks from CPUs. Modern graphics processors are now powerful enough to fastly perform rapid mathematical calculations for many different purposes other than image rendering.
A CPU consists of a few cores optimized for sequential serial processing. It is designed to maximize the performance of a single task within a job.

On the other hand, a GPU uses thousands of smaller and more efficient cores for a massively parallel architecture aimed at handling multiple functions at the same time.
Modern GPUs provide superior processing power and efficiency over their CPU counterparts regarding tasks that require multiple parallel processes, such as Deep Learning [28].
Considering the implemented MATLAB script, the use of multi-GPU parallel computing provided an impressive improvement on training process duration when compared to single GPU or CPU trainings. The results of various tests on training time are presented in Table 2.1, both for true color and grayscale datasets.

	4 GPUs	Single GPU	Single CPU
True Color Dataset	2 mins, 7 secs	39 mins, 09 secs	43 mins, 33 secs
Grayscale Dataset	1 min, 7 secs	30 mins, 69 secs	37 mins, 43 secs

Table 2.1: Training duration results obtained using 4 GPUs, a single GPU and a Single CPU, both regarding the true color and grayscale versions of the same dataset.

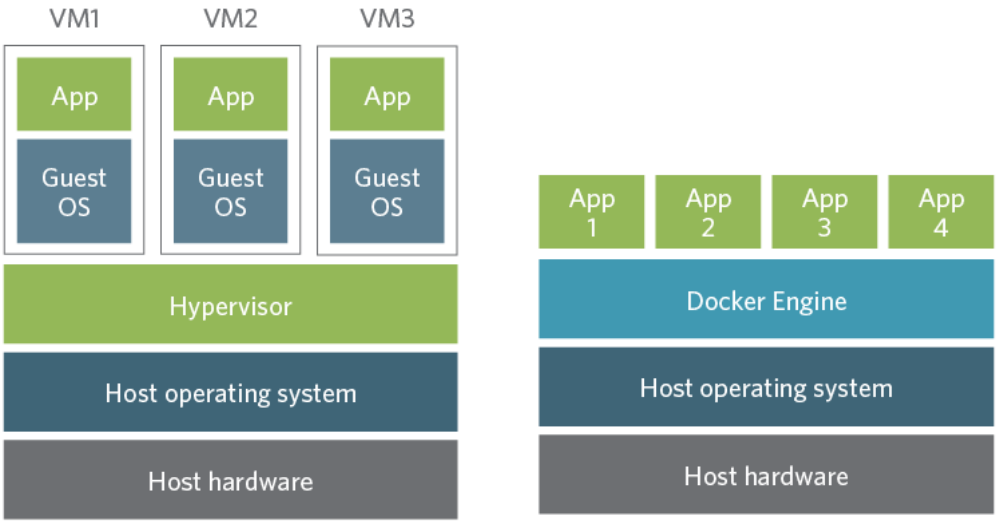


Figure 2.7: Virtual Machines (left panel) and Containers (right panel) [27].

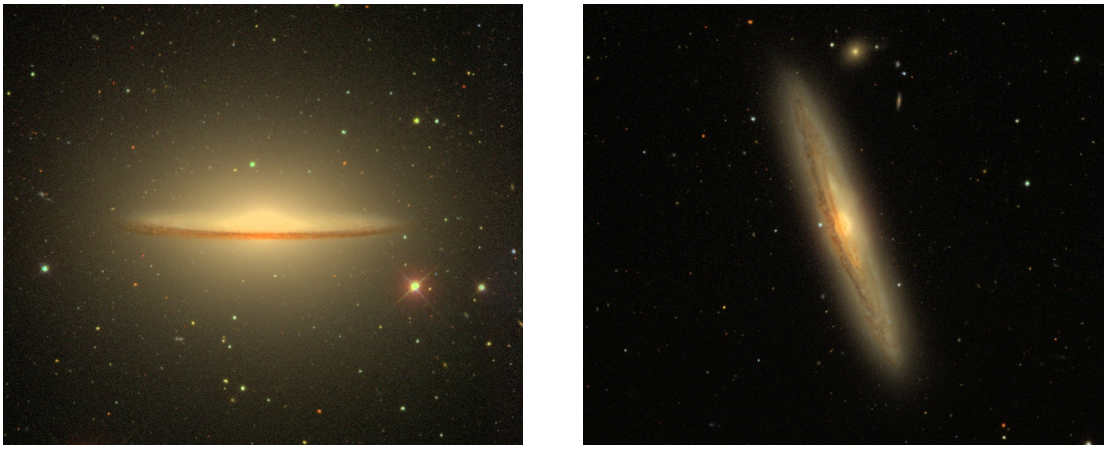
Chapter 3

Image Dataset Creation and Pre-Processing

In this chapter, the sample images selection process is described, focusing on the adopted criteria in this stage of the thesis work. Variants of image pre-processing routines are presented and discussed as well.

3.1 Image Dataset Selection

In order to get trained for galaxy morphology classification, Deep Learning algorithms needs a large amount of images. Following the Hubble Sequence's scheme, in this work the goal of the training was limited to be the classification of four different classes: 1. elliptical galaxies (E); 2. spiral galaxies (S); 3. barred spiral galaxies (SB); 4. Edge on galaxies ($EDGE\ ON$). The latter one includes galaxies of any type that are seen from the side (Figure 3.1), consequently no morphological assumption can be made regarding them.



(a) MESSIER 104 (b) NGC 4216

Figure 3.1: Two examples of edge-on galaxies.

Starting from scratch, some criteria have to be set in order to create a coherent dataset.
Galaxies are usually named and listed in catalogues, for example the Messier Catalogue or the New General Catalogue (NGC). Hence, the first step of the selection process is to find a suitable list of galaxies to search for. In order to do so, the NASA/IPAC Extragalactic Database ([1]NED) was used in order to sort and list galaxies by various morphological classifications.
After completing this stage, it was necessary to find a tool for image search, selection and export. A suitable solution was found in the Strasbourg Astronomical Data Center (CDS) CDS [30], that collects and distributes astronomical and related information worldwide.
CDS provides various tools:

a. The SIMBAD astronomical database [31], a world reference database for the identification of astronomical objects;

b. VizieR [32], the catalogue service for the CDS reference collection of astronomical catalogues and tables published in academic journals;

c. The Aladin [33] interactive software sky atlas for access, visualization and analysis of astronomical images from many catalogues, which include different wavelengths and qualities (Figure 3.2).

Figure 3.2: Quality comparison between two images of MESSIER 109: 3.2(a) was taken from the Digitized Sky Survey (DSS) NASA/IPAC [34], while 3.2(b) was taken from the Sloan Digital Sky Survey [4].

In order to implement galaxy morphological classification, only the visible part of the electromagnetic spectrum is relevant, so optical catalogues are preferred. Among the many different catalogues available in the CDS, the Sloan Digital Sky

[1]The NASA/IPAC Extragalactic Database (NED) is operated by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the NASA. [29]

Survey (SDSS), in all its releases, was chosen. [2]The SDSS is a major multi-spectral imaging and spectroscopic survey using a dedicated 2.5 metre wide-angle optical telescope at Apache Point Observatory in New Mexico, United States. SDSS's final imaging data release (DR9) covers over 35% of the sky, with photometric observations of around nearly 1 billion objects carried on from 1998 to 2009.
Images were taken using a photometric system of five filters (u, g, r, i and z). The telescope's imaging camera is made up of 30 CCD chips, arranged in 5 rows of 6 chips each row has a different optical filter, with average wavelengths of 355.1 nm for the u filter, 468.6 nm for the g, 616.5 nm for the r, 748.1 nm for the i and 893.1 nm for the z.
Hence, it is correct to assume that SDSS's images are real true color images and as such they are processed in this work.
In conclusion, the image selection was carried out manually, combining the features of different resources: the NASA/IPAC Extragalactic Database, the astronomical database SIMBAD, the Aladin interactive image visualization software (the last two were featured in the Strasbourg Astronomical Data Center) and the Sloan Digital Sky Survey.
The selection resulted in the creation of a database of 625 elliptical galaxies, 175 spiral galaxies, 322 barred spiral galaxies and 189 edge-on galaxies; the image format is [649 px ×550 px ×3].

3.2 Image Pre-processing

The images selected as schematized in the previous paragraph can be considered "raw images" featuring specific peculiarities regarding their background or the galaxy shape. Some of them may present other more distant galaxies in the background, or show a nearer star appearing as a rather bright cross-shaped object on the image. Sometimes another object is interposed between the galaxy and the observer, and this can lead to a distortion of the galaxy's shape.
These raw images have been used to train the network and will form Dataset 1, but some image pre-processing routines have been used in order to try to improve the obtained results:

a. **Sharpening High-Pass Gaussian Filter**

   In order to refine the galaxies' shape outline, a simple sharpening high pass filter was implemented.
   Considering the fact that the high-pass filter is equivalent to the identity matrix minus the low-pass filter, the MATLAB code line:

   ```
   h = padarray(2,[2 2]) - fspecial('gaussian',[5 5],2)
   ```

   was used to create the desired filter. The `fspecial` command creates a gaussian $[5 \times 5]$ 2-D low-pass filter; the `padarray` command creates a $[5 \times 5]$

[2]Gunn et al. [35]; Gunn et al. [36]; York et al. [37]; Frieman et al. [38]; Eisenstein et al. [39]

matrix setting a desired value in the central element and setting the value 0 for the remaining elements.
In order to preserve the brightness, the sum of the elements of the filter's matrix must be 1, so the central value of the padding matrix must be 2. For example:

$$h = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0.0232 & 0.0338 & 0.0383 & 0.0338 & 0.0232 \\ 0.0338 & 0.0492 & 0.0558 & 0.0492 & 0.0338 \\ 0.0383 & 0.0558 & 0.0632 & 0.0558 & 0.0383 \\ 0.0338 & 0.0492 & 0.0558 & 0.0492 & 0.0338 \\ 0.0232 & 0.0338 & 0.0383 & 0.0338 & 0.0232 \end{bmatrix} = \\ = \begin{bmatrix} -0.0232 & -0.0338 & -0.0383 & -0.0338 & -0.0232 \\ -0.0338 & -0.0492 & -0.0558 & -0.0492 & -0.0338 \\ -0.0383 & -0.0558 & 1.9368 & -0.0558 & -0.0383 \\ -0.0338 & -0.0492 & -0.0558 & -0.0492 & -0.0338 \\ -0.0232 & -0.0338 & -0.0383 & -0.0338 & -0.0232 \end{bmatrix} \tag{3.1}$$

Then, the `imfilter` command is used to apply the created filter to the image. These images filtered in this way constitute Dataset 2 (Figure 3.3).

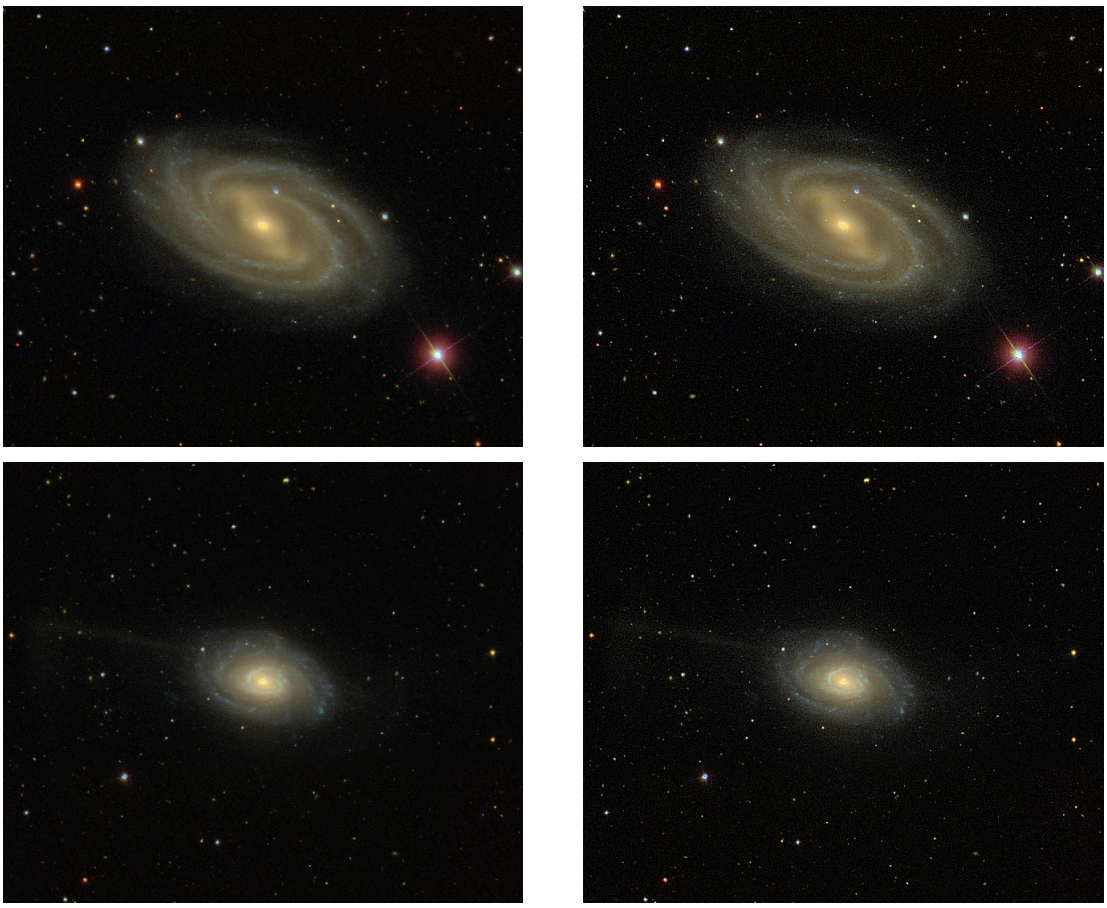


Figure 3.3: Comparison between the original images (left panels) and the ones obtained using the sharpening high-pass Gaussian filter (right panels) of MESSIER 109 and ESO 574-G028

b. **High-Pass FFT Gaussian Filter**

Another possible sharpening filter was implemented by the fast Fourier transform (FFT). The MATLAB command `fft2` was used to compute the two-dimensional FFT of the image. Considering the fact that true color images are being used, this process is applied to all the three two dimensional components of the images (Red, Green and Blue).
Then `fftshift` is used to shift the zero-frequency component to the center of the spectrum and this process is followed by the calculation of the Gaussian low-passing filter.
The same concept of the previous filter is applied: the high-pass filter is obtained by subtracting the low-pass filter to the identity.
The convolution between the transformed image and the filter is now computed and the inverse zero-frequency shift `ifftshift` is used.
The filtered image is finally obtained by the 2-D inverse fast Fourier transform `ifft2`. The images filtered in this way constitute Dataset 3 (Figure 3.4.

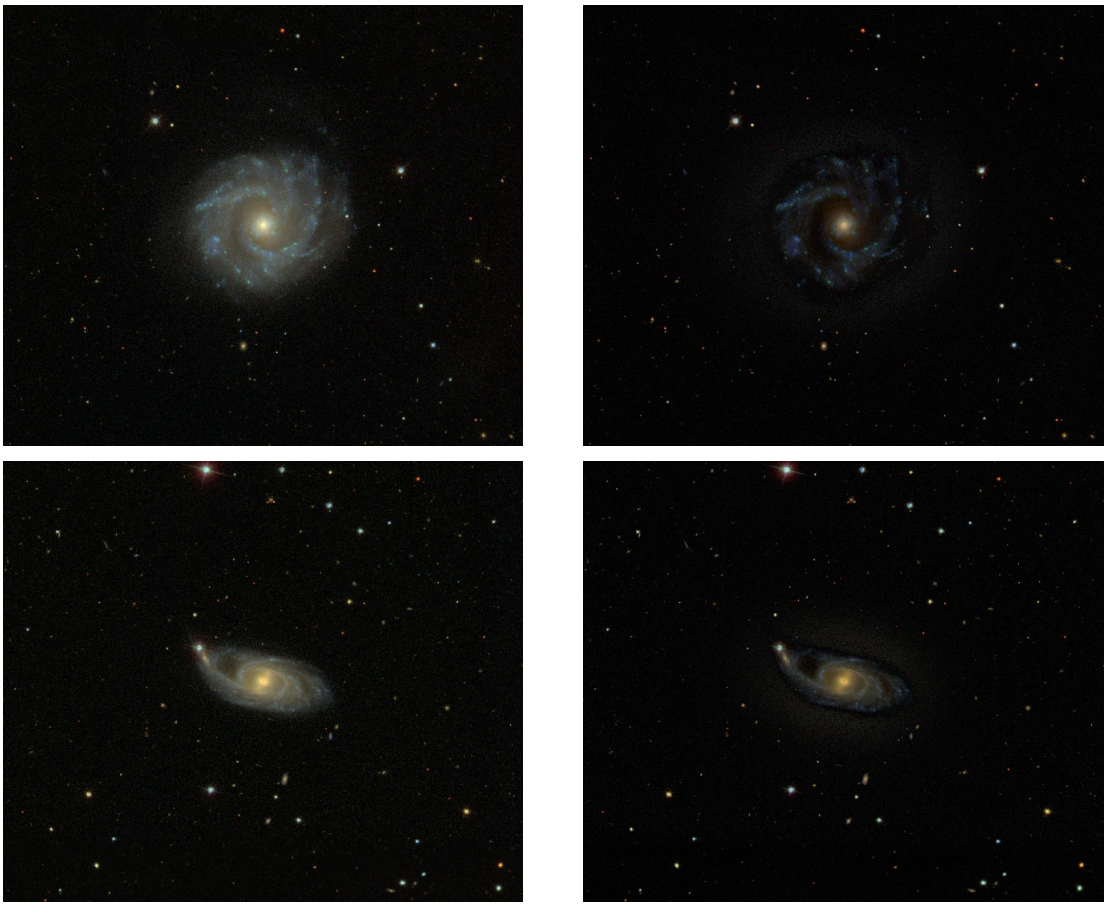


Figure 3.4: Comparison between the original images (left panels) and the ones obtained using the high-pass FFT Gaussian filter (right panels) of NGC 3631 and NGC 0151.

c. **Histogram Equalization and High-Pass FFT Gaussian Filter**

A possible solution to enhance the edges of shapes in the images is to increase contrast trough histogram equalization and then to apply the high-pass FFT Gaussian Filter.
An image histogram is a graphical representation of intensity distribution in a digital image. Using the `histeq` command, histogram equalization is obtained, as displayed in Figure 3.5.
The resulting image is processed through the previous script, the results are displayed in Figure 3.7. These images will form Dataset 4.

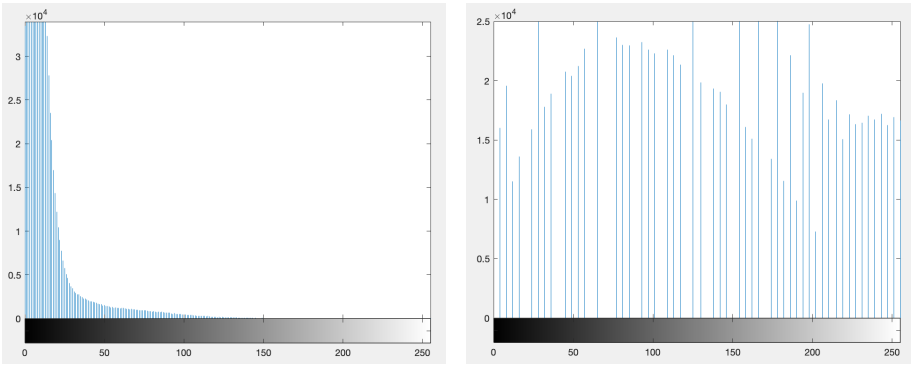


Figure 3.5: The histogram of an image before and after equalization.

d. **Grayscale Conversion**

The colortype conversion from true color to grayscale (Figure 3.6) simplifies the images, resulting in lower computational effort.
This conversion is achieved using the MATLAB command `rgb2gray`.

Figure 3.6: The true color and grayscale versions of NGC 2683.

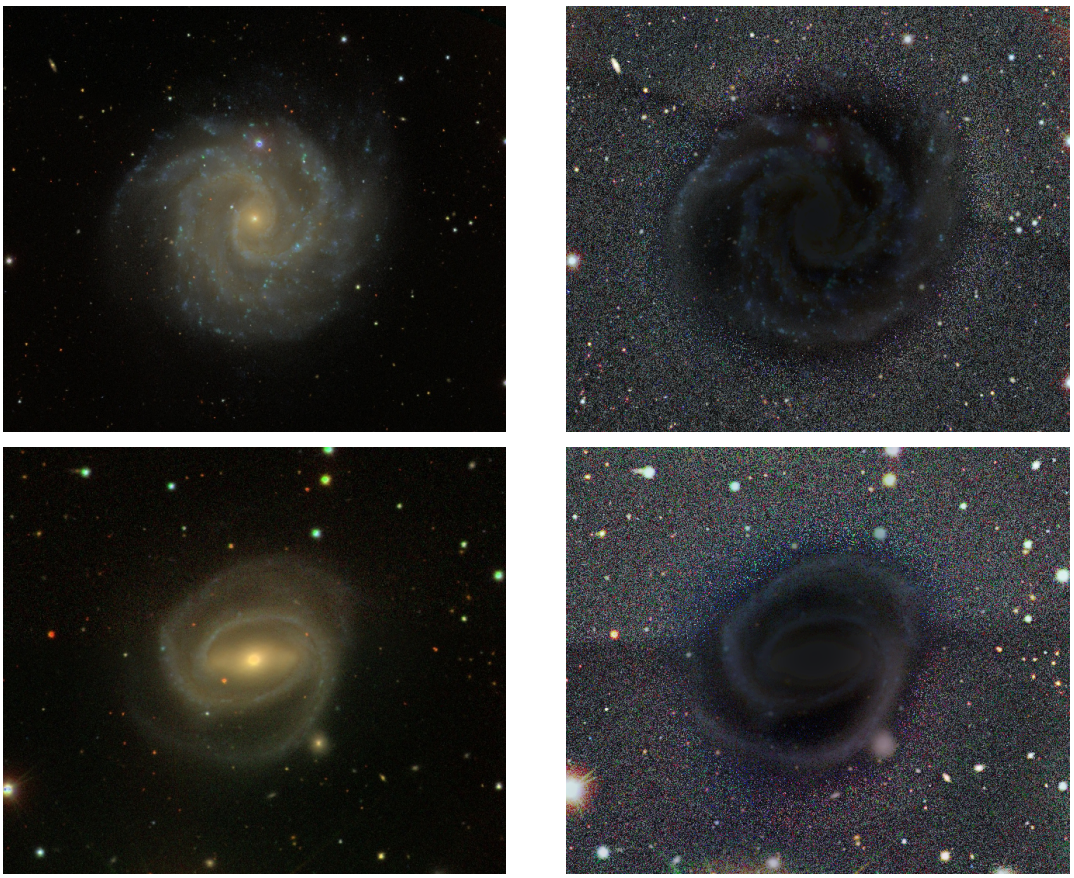


Figure 3.7: Comparison between the original images (left panels) and the ones obtained using histogram equalization and the high-pass FFT Gaussian filter (right panels) of NGC 3184 and NGC 0266.

Chapter 4

Training Process Results and Analysis

In this chapter, the results of the different training processes will be discussed, with particular attention to the relevant diagnostic plots obtained in the various cases.
For each dataset, both the true color and the grayscale images were considered. Furthermore, two different training processes will be presented in each paragraph:

1. The first one for classification of four galaxy classes (elliptical, spiral, barred spiral an edge on);

2. The second one regarding the recognition of three galaxy classes (spiral, barred spiral an edge on).

Then, the result of the mean of 50 runs for each training set will be discussed.
When looking at the plots in the following sections, the reader should refer to the reference key depicted in Figure 4.1:

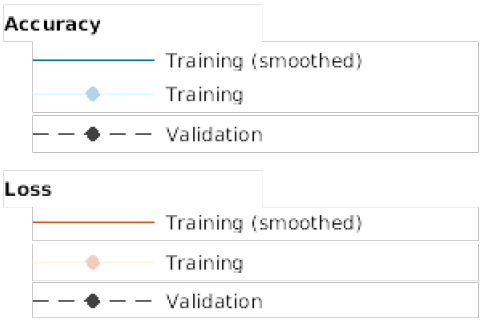


Figure 4.1: Reference key for the diagnostic plots.

4.1 Results of the Modified CNN Architecture

As mentioned in Section 2.4.3, an additional Max-Pooling Layer was added to the CNN's architecture, and the different results of the two training processes are reported in Figure 4.2.

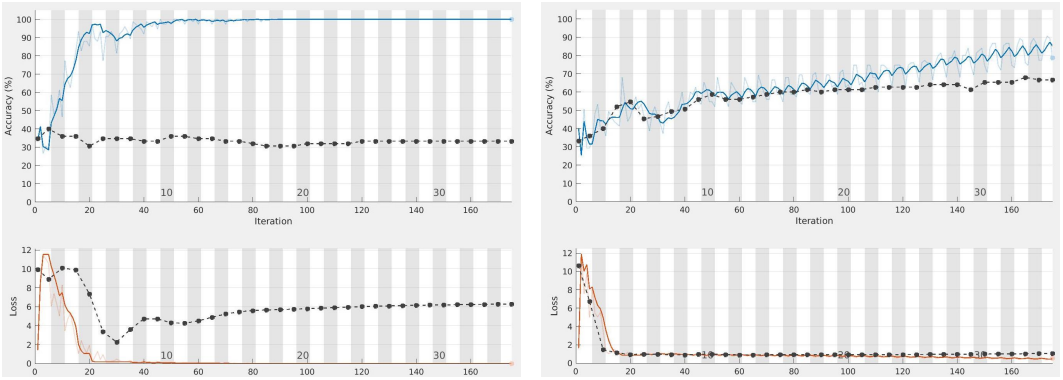


Figure 4.2: Diagnostic plots of two training processes regarding the same training and validation datasets. The one on the left panel results from the layer layout of the original MATLAB template, it provides an internal accuracy of 33.33% and an externally tested precision of 36%. The one on the right panel results from the modified layer layout and has an internal accuracy of 66.67% and an externally tested precision of 56%.

The better results of the modified CNN can be explained by analysing the layer layout, schematized in Figure 4.3.
Looking at the original MATLAB architecture, it appears that the second "convolutional block" isn't followed by a max-pooling layer before the fully connected layer. Usually, the fully connected layer follows a max-pooling layer: this was the first clue that lead to the idea of a possible modification of the architecture.

1. Input Layer
2. Convolutional Layer
3. Batch Normalization Layer — Convolutional Block
4. ReLU Layer
5. Max-Pooling Layer
6. CConvolutional Layer
7. Batch Normalization Layer — Convolutional Block
8. ReLU Layer
9. Fully Connected Layer
10. Softmax Layer
11. Classification Layer

1. Input Layer
2. Convolutional Layer
3. Batch Normalization Layer — Convolutional Block
4. ReLU Layer
5. Max-Pooling Layer
6. CConvolutional Layer
7. Batch Normalization Layer — Convolutional Block
8. ReLU Layer
9. Max-Pooling Layer
10. Fully Connected Layer
11. Softmax Layer
12. Classification Layer

Figure 4.3: The two different layer layouts, on the left panel the original architecture provided by MATLAB, on the right panel the modified architecture.

This modification is justified by the fact that the max-pooling layer removes redundant information providing simpler data to the fully connected layer, which will then identify the different classification categories. Therefore, simpler information will provide a better classification and, thus, better results.
This kind of argument is quite general and may be applied to all kind of datasets. However, some tests have been run on the MNIST dataset and have provided no significant differences in training results. Hence, a conclusion can be drawn: for bigger and simpler datasets this kind of adjustment leads to no noticeable improvement of the results, but provides key benefits when working with smaller and more complex datasets.

4.2 Training Results for Dataset 1

In this section, the results obtained using the original images are presented, both for the true color and the grayscale datasets.

4.2.1 Dataset 1 True Color

Four Classes

Some typical obtained training diagnostic plots are reported in Figure 4.4.
Comparing Figures (a) and (b) with (c) and (d) and referring to Figure 2.6 as an ideal plot, one could guess that (a) and (b) provide better results because the training plot for accuracy (blue line) asymptotically goes to 100%, but actually the complete opposite is true.
With regard to accuracy, when the trend of the training plot goes asymptotically to 100% but is not followed by a validation plot with the same behaviour (as happens in Figure 2.6), it means that the statistical model is overfitting.
Another clue of overfitting is identifiable when the information loss plot asymptotically goes to zero, but is not followed by a validation plot with a similar trend.
Considering Figures (c) and (d) instead, it's clear that the training and the validation plots have about the same behaviour and tend to similar percentages.
In conclusion, the Deep Learning training (a) and (b) learnt to classify only the exact images provided from the training dataset, but could not extract the parameters to build a mathematical model able to fit to external data. Indeed, training (a) and (b) have an accuracy of 31% and 35% but, when tested with an external dataset, both provide an accuracy of 22.67%. This means that randomly guessing one of the four categories could result in a better outcome compared to using the network for classification.
On the other hand, training (c) and (d) have built a mathematical model that has adapted to the dataset and can fit to external information. In fact, training (c) and (d) have an accuracy of 38% and 40% but, when tested with an external dataset, provide an accuracy of 58.67% and 54.67% respectively.

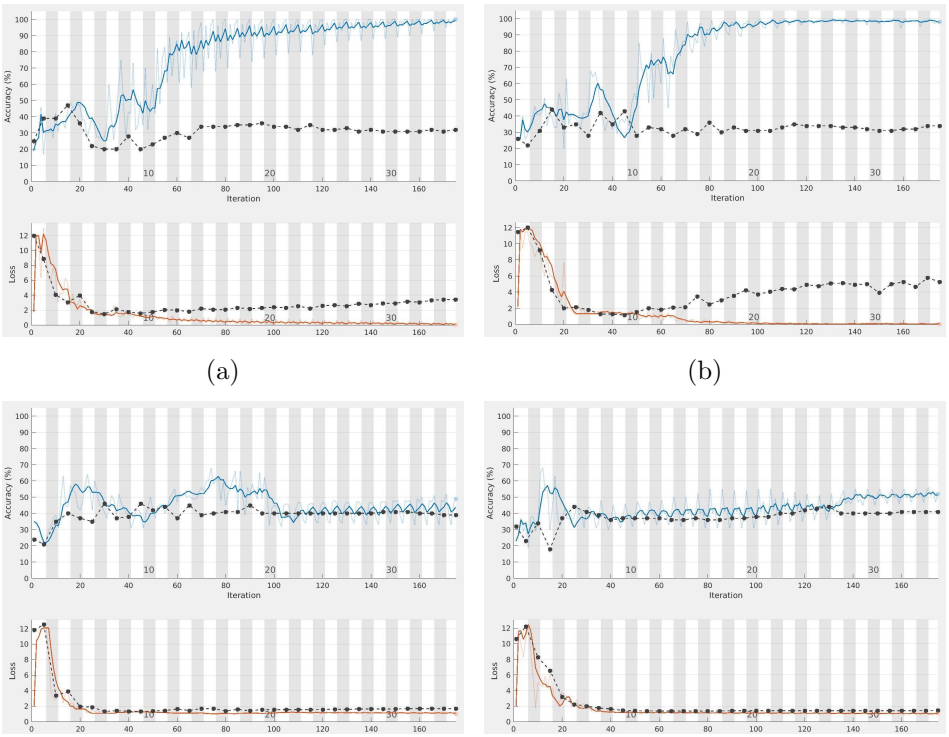


Figure 4.4: Diagnostic plots obtained using Dataset 1 for the classification of four different types of galaxies.

Three Classes

Similar considerations about overfitting can be made for Figure 4.5(a), although this training provides actually good results: the accuracies are 50.67% and the external test resulted in a correct classification 48% of the time.
Training (b) resulted in an accuracy of 57.33% for the internal validation but provided a slightly worse result when tested externally: 54.67%.

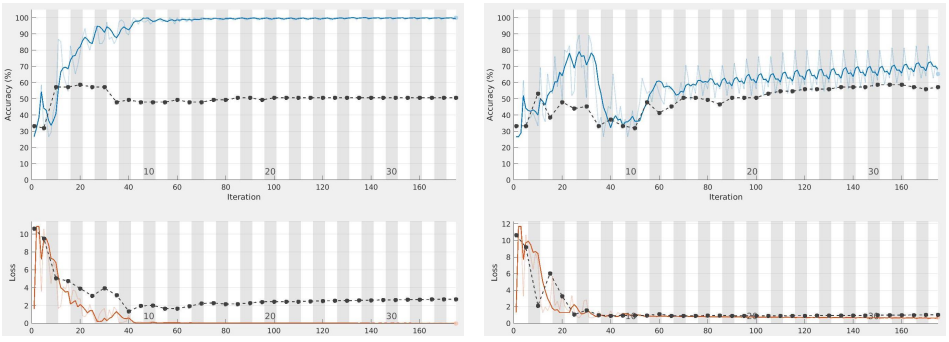


Figure 4.5: Diagnostic plots obtained using Dataset 1 for the classification of three different types of galaxies.

4.2.2 Dataset 1 Grayscale

Four Classes

Considering Figure 4.6, the plots of training (a) and (b) display a typical overfitting behaviour, the accuracy is 41% and 31% respectively, with precision of 25.33% and 16% when tested with an external dataset. These results are quite unsatisfactory, being worse than random guessing.
Training (c) and (d) provide good results: the accuracy is 36% and 35%, but the external test resulted in a well-trained model with accuracy of 53.33% and 45.33% respectively.

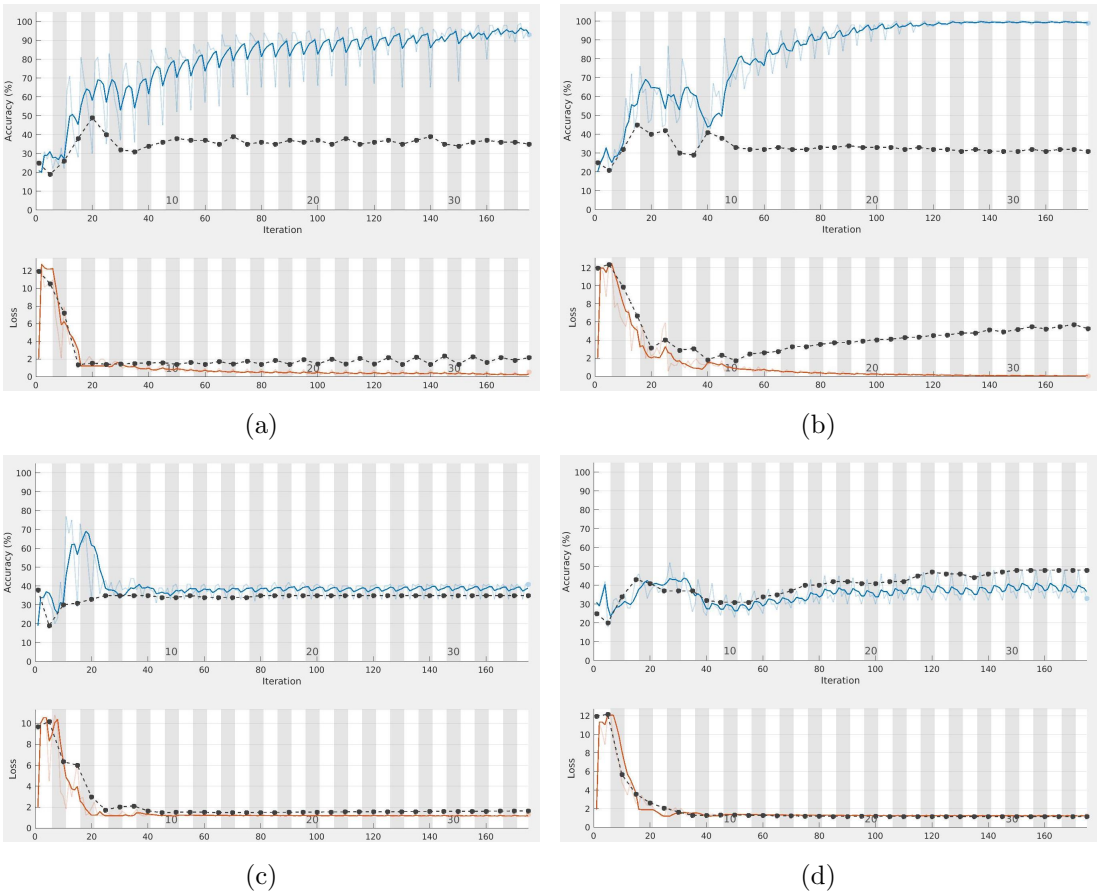


Figure 4.6: Diagnostic plots obtained using the grayscale version of Dataset 1 for the classification of four different types of galaxies.

Three Classes

The usual considerations about overfitting can be made for plots (a) and (b) in Figure 4.7. The accuracy results are actually good, with 49.33% and 40% but when put into testing the network doesn't perform as expected, providing a precision of 38.67% and 33.33%.
Training (c) resulted in an accuracy of 58.67%, but in a slightly worse result when tested: 52%; on the other hand, training (d) behaved exactly as expected, proving both accuracy and test precision of 46.67%.

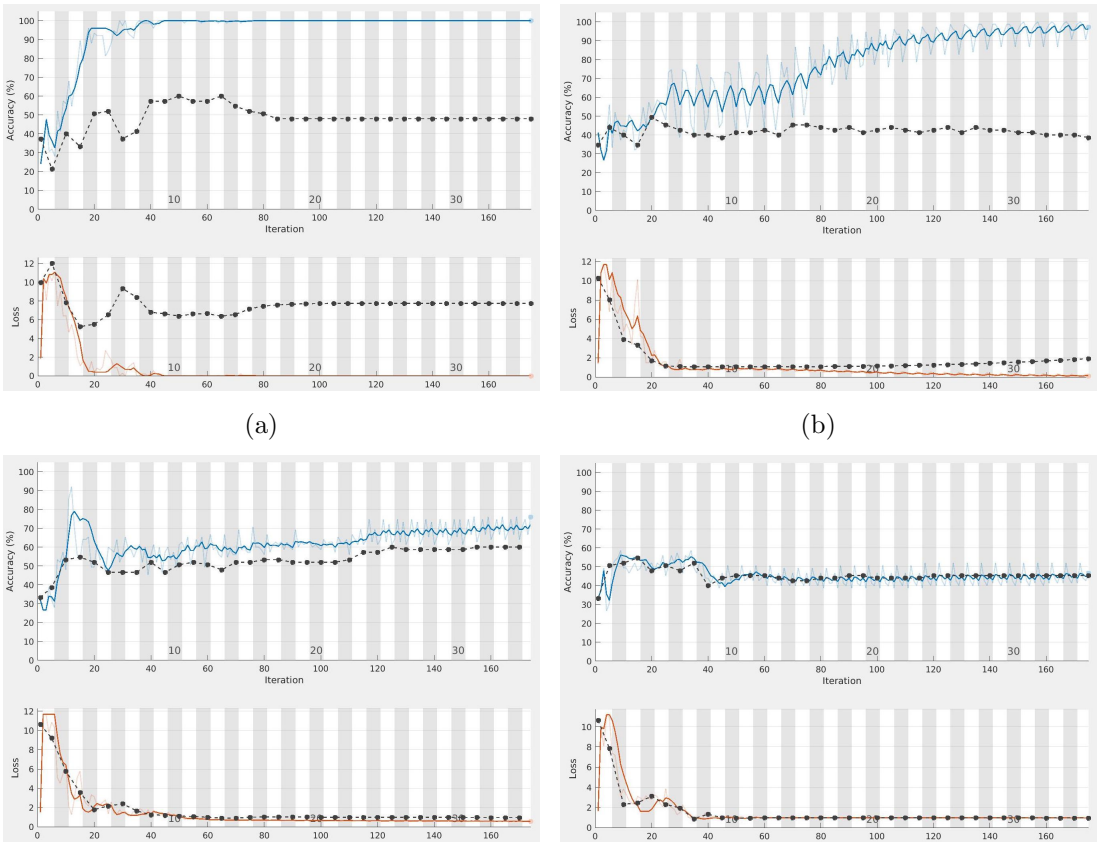


Figure 4.7: Diagnostic plots obtained using the grayscale version of Dataset 1 for the classification of three different types of galaxies.

4.3 Training Results for Dataset 2

In this section, the results obtained using images filtered by a Sharpening High-Pass Gaussian Filter are presented, relevant both to true color and grayscale datasets.

4.3.1 Dataset 2 True Color

Four Classes

In Figures 4.8 (a) and (b) a typical overfitting behaviour is depicted, the accuracies are of 32% and 30.67% respectively, but the models slightly underperform when tested by external data, resulting in an accuracy of 30.67% and 32% respectively. Plots (c) and (d) provide better results, with an accuracy of 37% and 38%, when externally tested the networks provide good classification outcomes with accuracy of 42.67% and 48%.

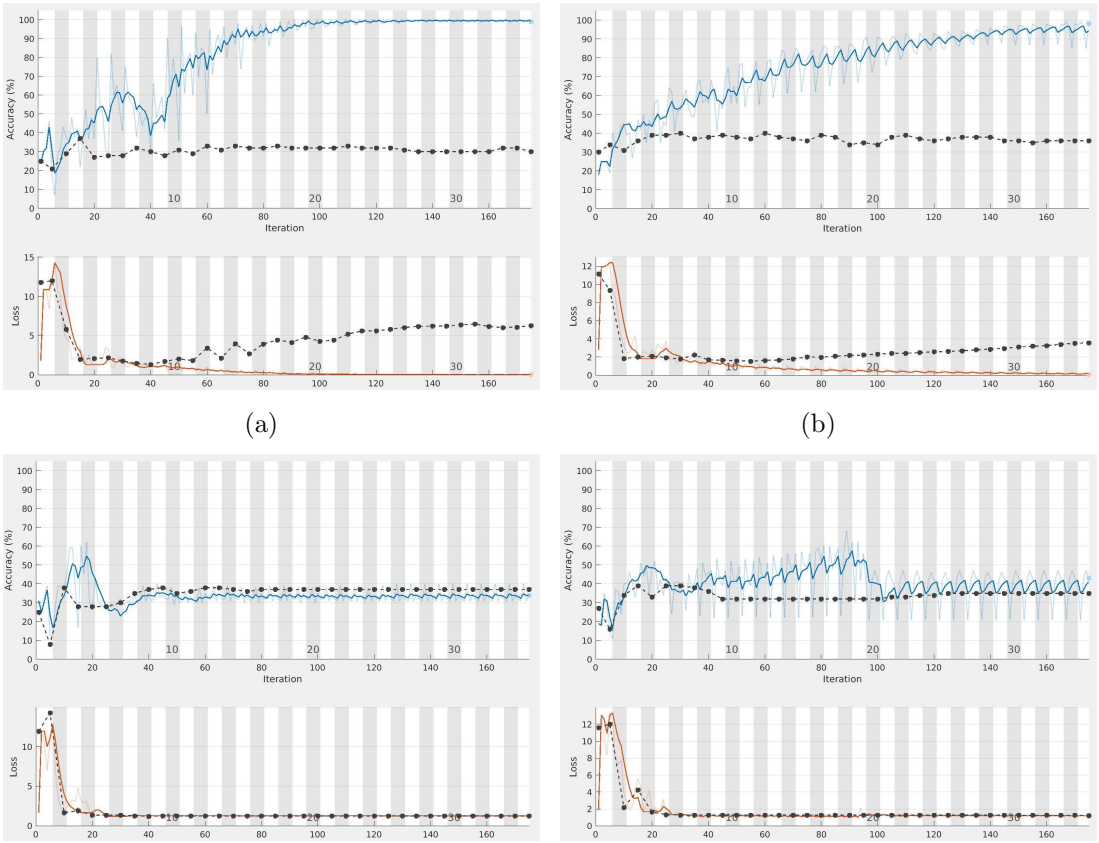
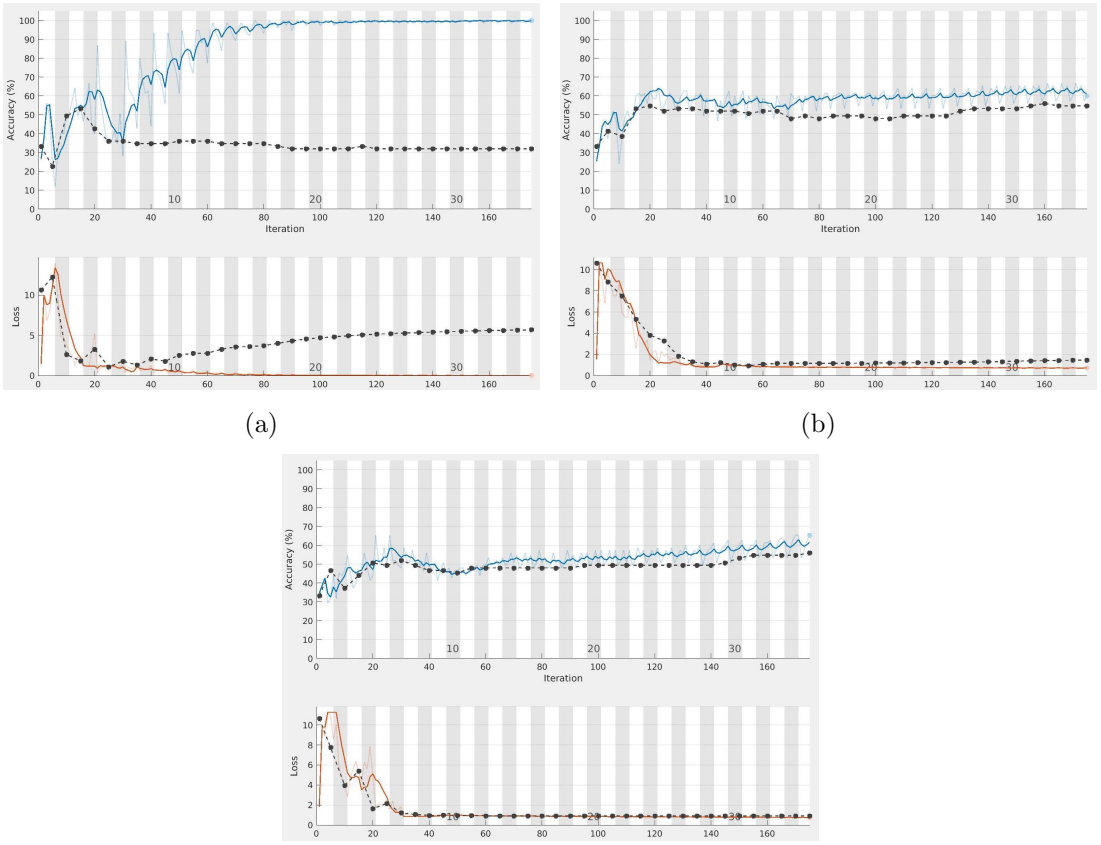


Figure 4.8: Diagnostic plots obtained using Dataset 2 for the classification of four different types of galaxies.

Three Classes

Figure 4.9 (a) reports the results of an overfitted training: accuracy is 33.33% and an external test provides a slightly better result of 36%.
Training (b) and (c) provided an accuracy of 53.33% and 54.67% resulting in a well-trained model, both the external tests resulted in an accuracy of 58.67%.



(a) (b)

(c)

Figure 4.9: Diagnostic plots obtained using Dataset 2 for the classification of three different types of galaxies.

4.3.2 Dataset 2 Grayscale

Four Classes

Figure 4.10 (a) depicts the usual overfitting behaviour, the accuracy is 36% and the external test results in a slightly improvement with an accuracy of 40%.
Plot (b) has an intermediate behaviour, probably if the number of epochs had been increased it would have tended to overfitting, in fact the internal accuracy is 46% and the external is 44%.
Both training (c) and (d) provide good results: with an accuracy of 34% each and an external test precision of 42.67% and 46.67% respectively.

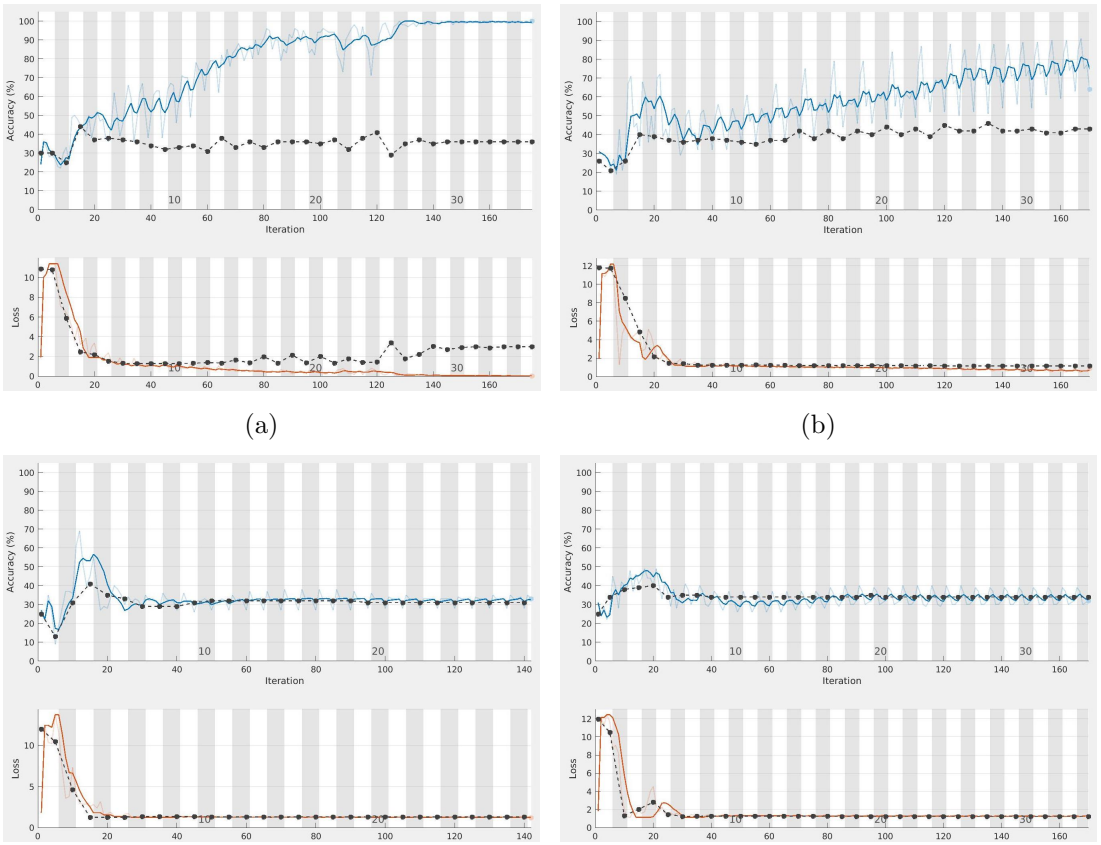
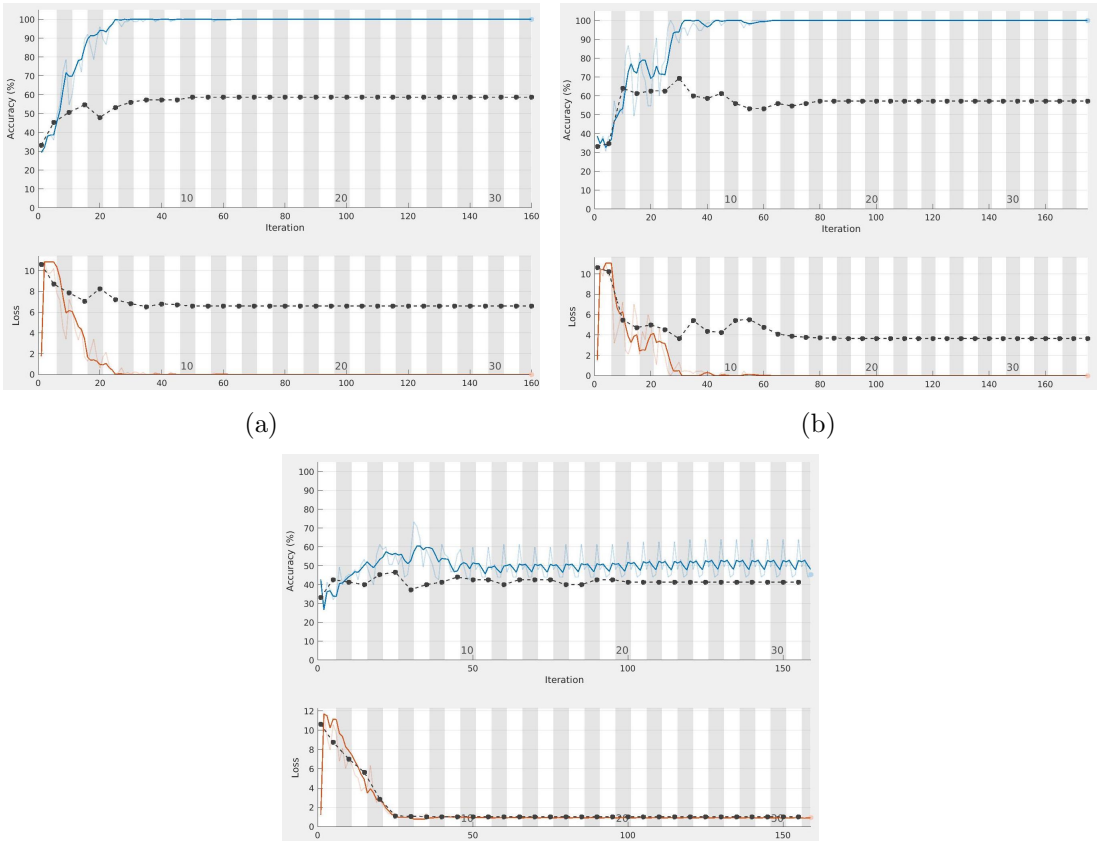


Figure 4.10: Diagnostic plots obtained using the grayscale version of Dataset 2 for the classification of four different types of galaxies.

Three Classes

Figures 4.11 (a) and (b) both exhibit an overfitting behaviour, although both of them result in good performances, with an accuracy of 58.67% and 57.33% respectively, and an external test precision of 52.00% and 58.67%.
Plot (c) depicts good training results, but the accuracy is 41.33% and the external test precision is 46.67%. Unexpectedly, despite the overfitting behaviour, the first two trainings have provided better results.



(a) (b)

(c)

Figure 4.11: Diagnostic plots obtained using the grayscale version of Dataset 2 for the classification of three different types of galaxies.

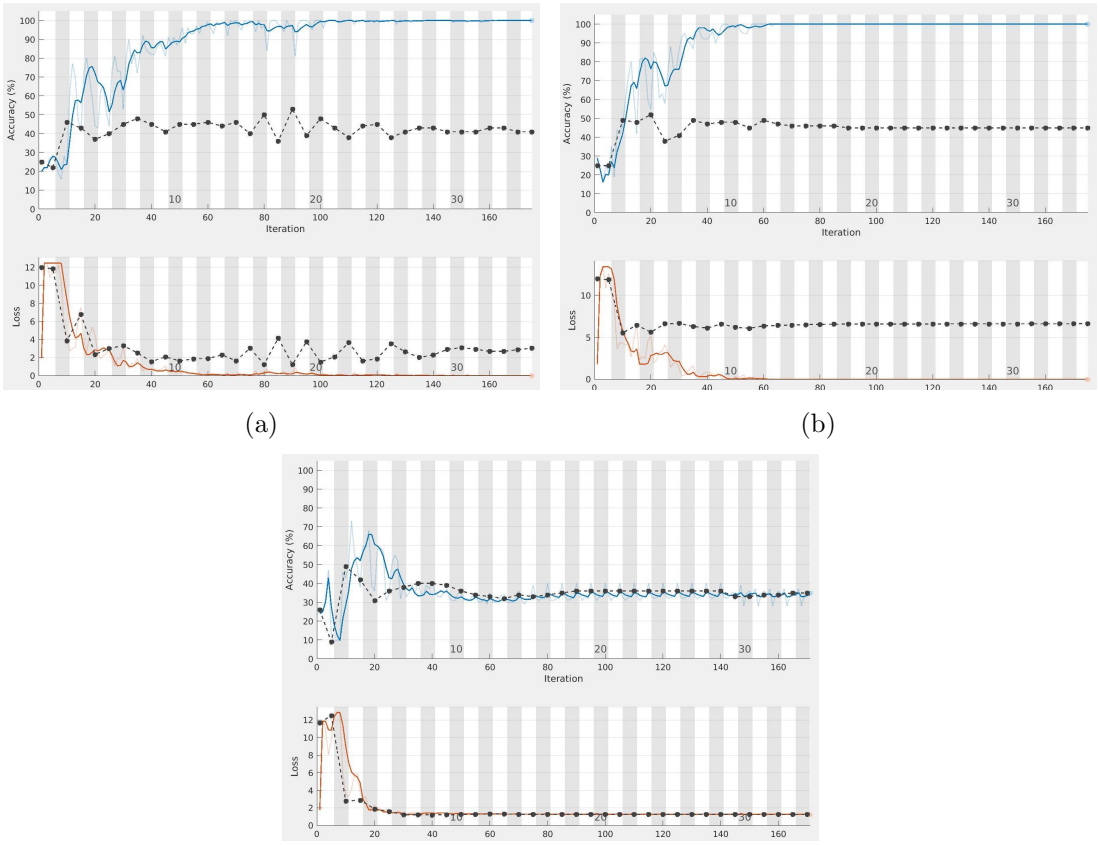
4.4 Training Results for Dataset 3

In this section, the results obtained using the High-Pass FFT Gaussian filtered images are presented, both for the true color and the grayscale images.

4.4.1 Dataset 3 True Color

Four Classes

Figures 4.12 (a) and (b) both emphasize an overfitting behaviour, the accuracy is 38% and 47% respectively; unexpectedly both networks provided good results when tested externally, with an accuracy of 44% and 52%.
Plot (c) reports good training results with an accuracy of 35% and a good performance when tested externally, with a precision of 48%.



(a) (b)

(c)

Figure 4.12: Diagnostic plots obtained using Dataset 3 for the classification of four different types of galaxies.

Three Classes

Considering Figure 4.13, plots (a) and (b) depict a typical overfitting behaviour; (a) has a promising accuracy of 62.67%, but the percentage drastically drops when the network is externally tested, resulting in an accuracy of 33.33%. Training (b) does slightly better, with an accuracy of 68% and a testing precision of 49.33%. Plot (c) resulted in good training results, the accuracy is 56% and the external test provided a positive result, with an accuracy of 61.33%.

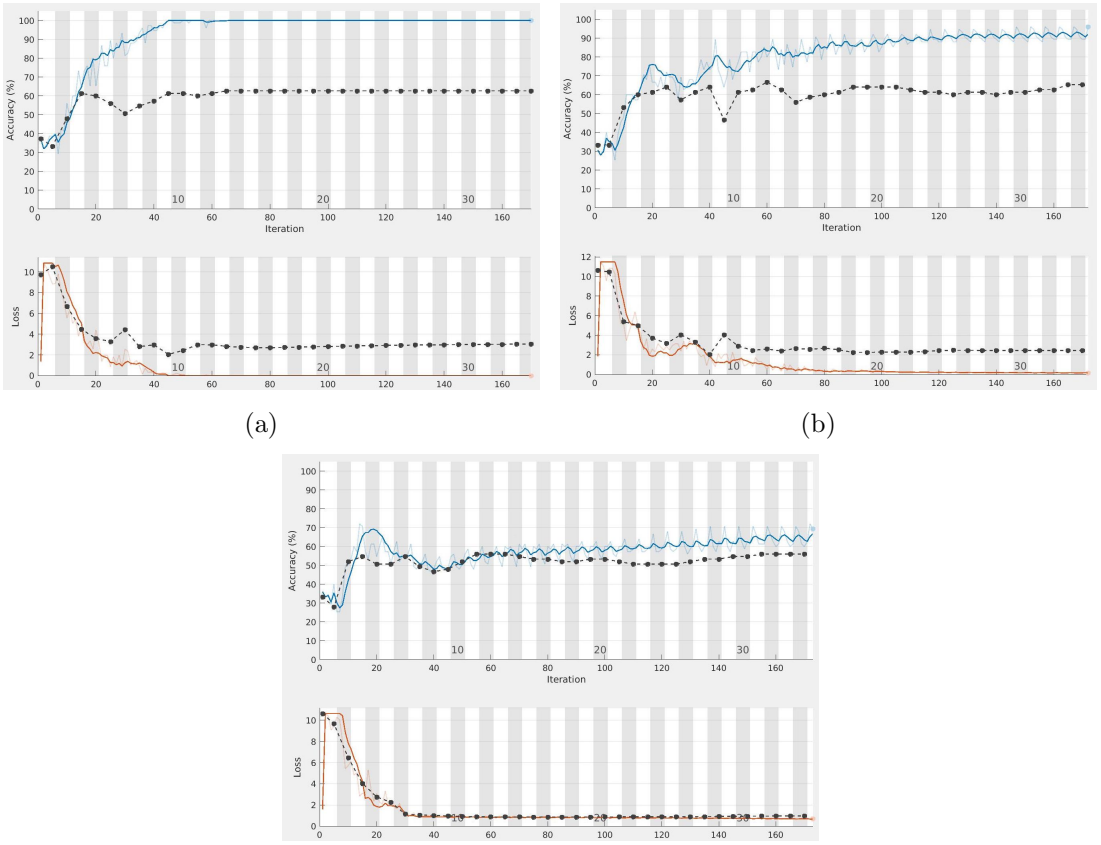
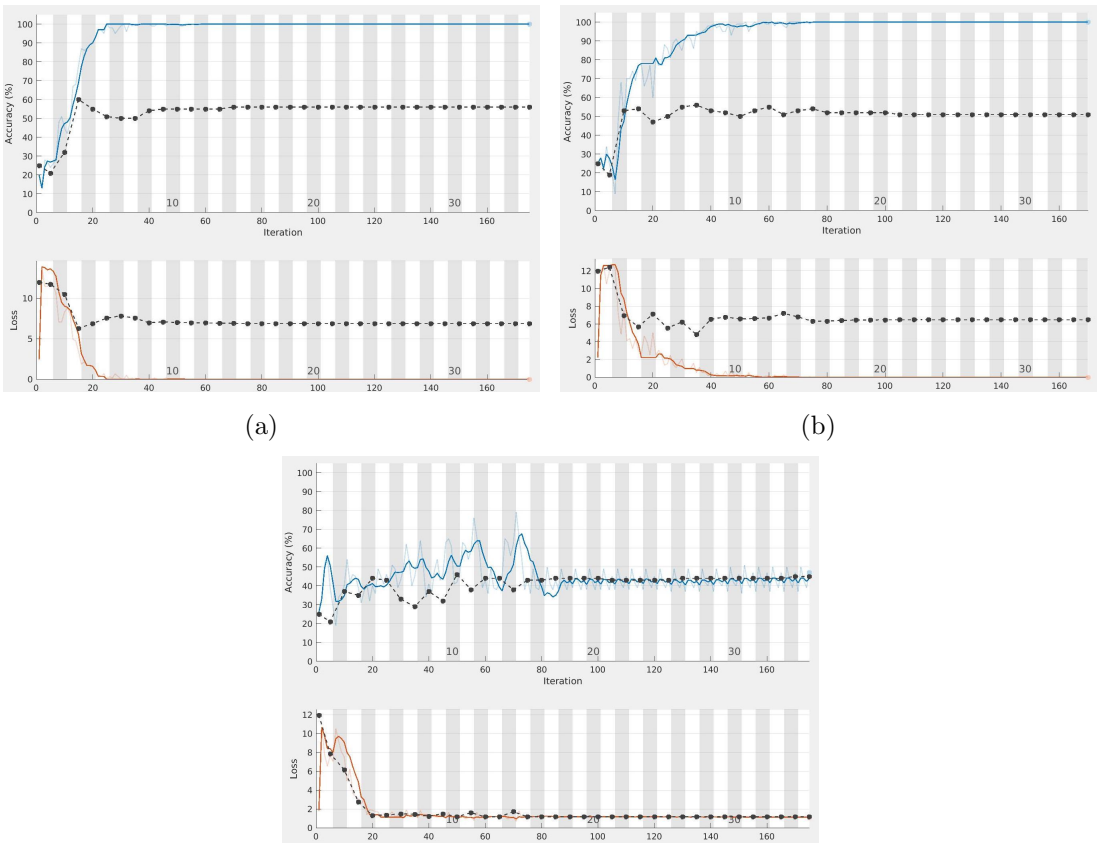


Figure 4.13: Diagnostic plots obtained using Dataset 3 for the classification of three different types of galaxies.

4.4.2 Dataset 3 Grayscale

Four Classes

Figures 4.14 (a) and (b) exhibit the typical overfitting behaviour: the accuracy results are pretty good with 56% and 52% respectively but the external testing results provide an accuracy of 49.33% and 38.67%.
Plot (c) depicts good training results, with an internal accuracy of 43% and an external testing accuracy of 52%.



(a) (b)

(c)

Figure 4.14: Diagnostic plots obtained using the grayscale version of Dataset 3 for the classification of four different types of galaxies.

Three Classes

Figures 4.15 (a) and (b) exhibit an overfitting training, the accuracy results are really good, being 69.33% and 68% respectively but the external testing provided an accuracy of 58.67% and 56%.
Plots (c) and (d) are representative of good training trends: both networks achieved an accuracy of 53.33%, resulting in an external testing accuracy of 66.67% and 58.67% respectively.

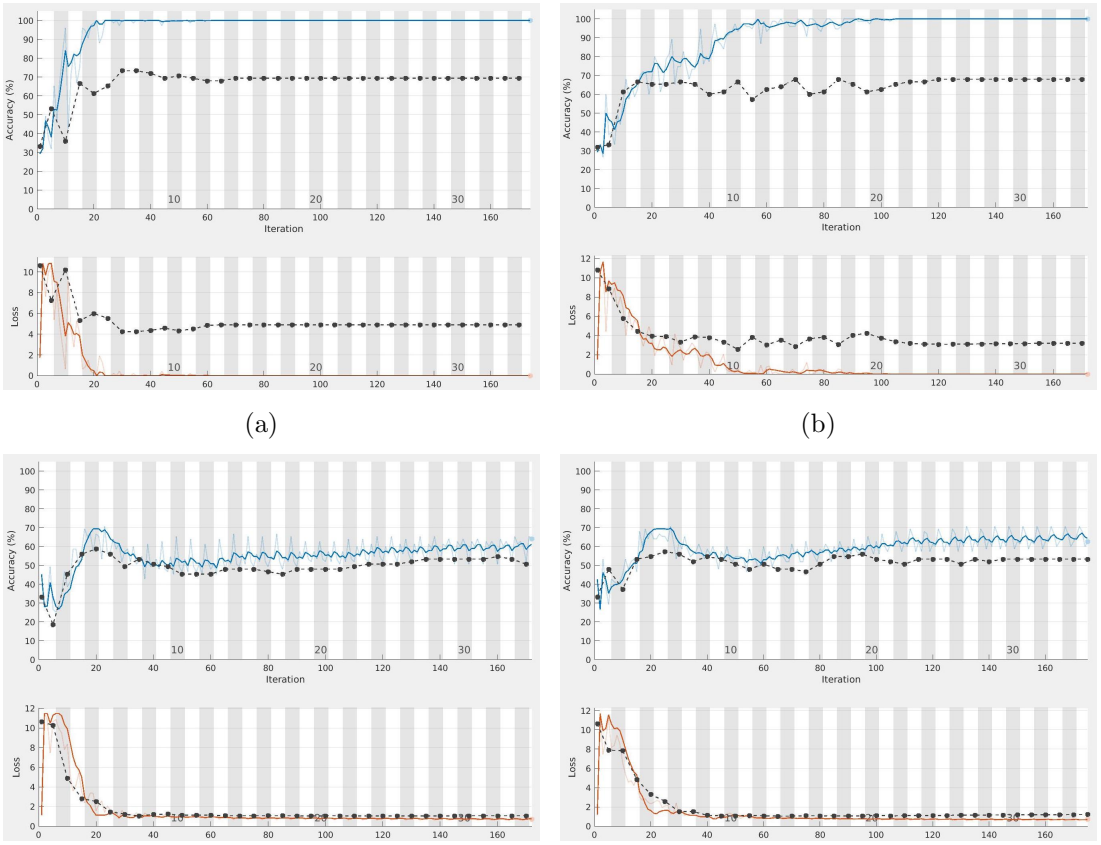


Figure 4.15: Diagnostic plots obtained using the grayscale version of Dataset 3 for the classification of three different types of galaxies.

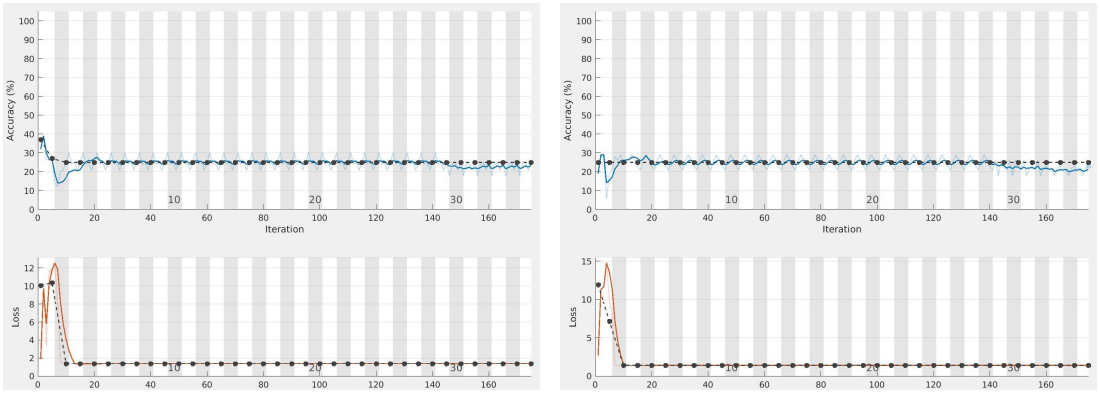
4.5 Training Results for Dataset 4

In this section, the results obtained using images processed by histogram equalization combined with the High-Pass FFT Gaussian filter are presented, both for true color and grayscale datasets are considered.

4.5.1 Dataset 4 True Color

Four Classes

Figures 4.16 (a) and (b) emphasize a strange trend: the training processes resulted in the same results as random probability, providing an accuracy of 25%. This behaviour was common to most of the plots produced using the true color version of dataset 4.
When externally tested the networks provided slightly better results, with an accuracy of 34.67% and 32%.



(a) (b)

Figure 4.16: Diagnostic plots obtained using Dataset 4 for the classification of four different types of galaxies.

Three Classes

Figures 4.17 (a) and (b) exhibit the same behaviour described in the previous paragraph; in this case, due to the fact that we are only considering three galaxy classes, the accuracy is 33.33%, the same as the random guessing probability. When externally tested, the networks provide some interesting results: the accuracy doesn't change, meaning that these training processes resulted in a failure. Plot (c) provides a better accuracy of 44%, but the external test resulted in almost the same behaviour as the previous trainings, with an accuracy of 34.67%.

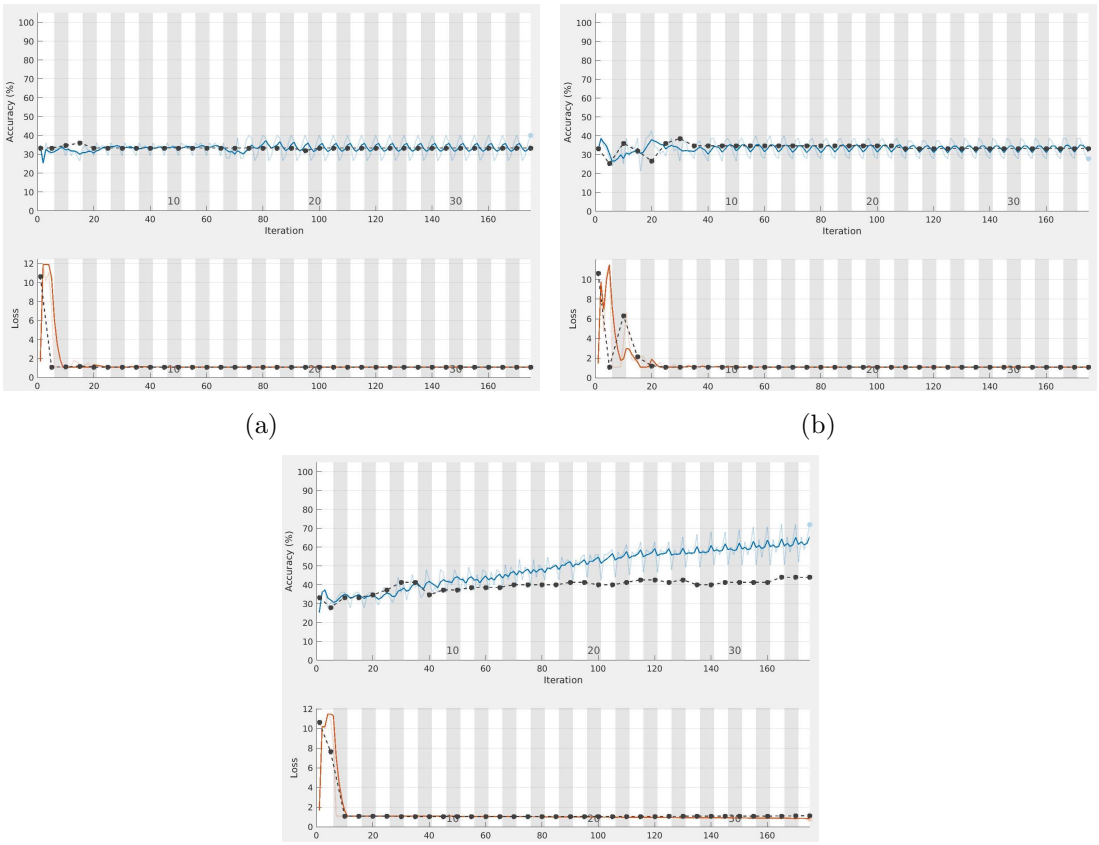


Figure 4.17: Diagnostic plots obtained using Dataset 4 for the classification of three different types of galaxies.

4.5.2 Dataset 4 Grayscale

Four Classes

The plots in Figure 4.18 depict a behaviour similar to the ones discussed in the first part of Section 4.5.1 (Figure 4.16): the accuracy is 29% for plot (a), 25% for plot (b) and 28% for plot (c).
These results are all located about the random guessing probability, but when externally tested the networks provided a slightly improved accuracy, with percentages of 37.33%, 33.33% and 40% respectively.

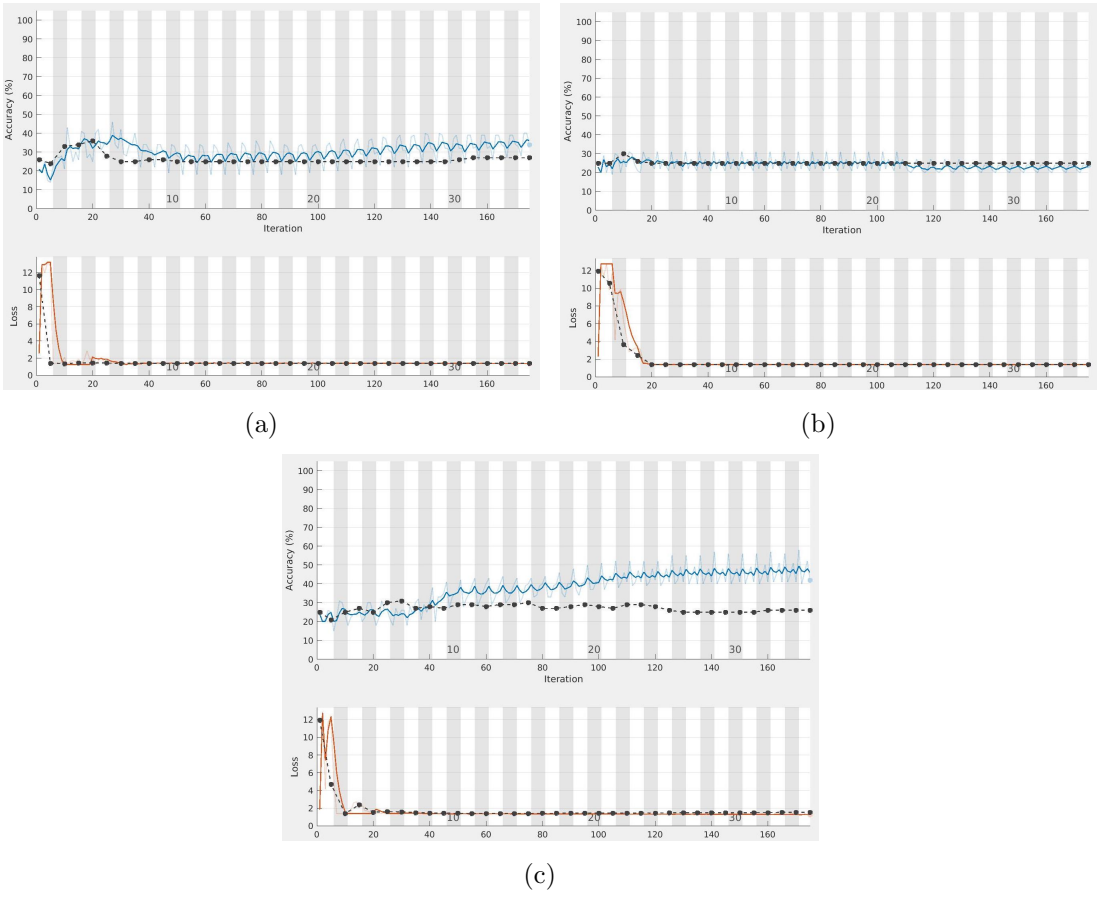


Figure 4.18: Diagnostic plots obtained using the grayscale version of Dataset 4 for the classification of four different types of galaxies.

Three Classes

The plots in Figure 4.19(a) report a behaviour similar to the ones discussed in the second part of Section 4.5.1 (Figure 4.17): the accuracy is 33.33% for plot (a), 36% for plot (b) and 40% for plot (c). These result are slightly better than the corresponding ones in Section 4.5.1, but the results of the external test provided similar disappointing outcomes, with precision of 33.33%, 33.33% and 30.67%.

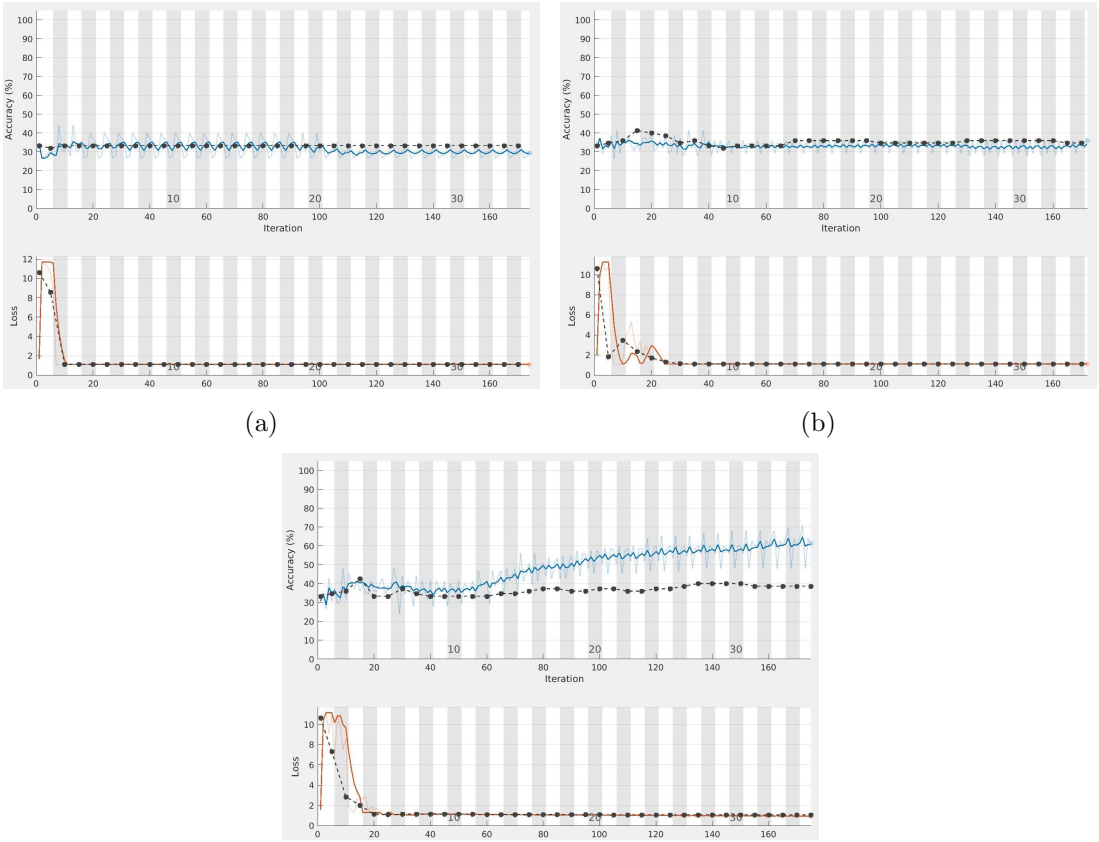


Figure 4.19: Diagnostic plots obtained using the grayscale version of Dataset 4 for the classification of three different types of galaxies.

4.6 Results of 50 Runs

In order to give an overview on the efficiency of the Deep Learning training, 50 runs were carried out for each of the eight datasets. The outcomes, given as the mean of the 50 values, are reported in Table 4.1.
Table 4.2 shows the percentage variation of each outcome, referring to the random guessing probabilities: 25% for 4 classes and 33.33% for three classes.

	4 Classes		3 Classes	
	Accuracy	External Test	Accuracy	External Test
Dataset 1 True Color	36.10	33.92	47.52	49.07
Dataset 1 Grayscale	36.10	35.78	48.29	47.25
Dataset 2 True Color	36.76	39.31	49.87	50.29
Dataset 2 Grayscale	36.40	37.73	49.81	49.52
Dataset 3 True Color	42.08	39.28	56.40	57.44
Dataset 3 Grayscale	43.68	42.64	56.69	55.95
Dataset 4 True Color	25.80	28.40	33.87	33.54
Dataset 4 Grayscale	26.54	26.80	33.84	33.89

Table 4.1: Outcomes of the mean of 50 runs for each of the training datasets, regarding both 4 classes and 3 classes training.

	4 Classes		3 Classes	
	Accuracy	External Test	Accuracy	External Test
Dataset 1 True Color	44.40	35.68	42.57	47.22
Dataset 1 Grayscale	44.40	43.12	44.88	41.76
Dataset 2 True Color	47.04	57.24	49.62	50.89
Dataset 2 Grayscale	45.60	50.92	49.44	48.57
Dataset 3 True Color	68.32	57.12	69.22	72.34
Dataset 3 Grayscale	74.72	70.56	70.09	67.87
Dataset 4 True Color	3.20	13.60	1.62	0.63
Dataset 4 Grayscale	6.16	7.20	1.53	1.68

Table 4.2: Percentage variation of the outcomes of each training, referred to the random guessing probabilities: 25% and 33.33%.

4.6.1 Interpretation of the Results

Looking at the results reported in Tables 4.1 and 4.2 the following conclusions can be drawn:

1. In general, there is no significant difference between the true color and the grayscale version of datasets 1, 2 and 4;

2. Although having similar accuracy results, both in three and four classes classification, Dataset 2 generally provides better results than Dataset 1 when under an external validation test; hence the Sharpening High-Pass Gaussian Filter improves the suitability of the dataset for morphological analysis;

3. Dataset 3 achieves better results than datasets 1 and 2, both when three and four classes classification is carried out.
   The difference between the outcomes of the true color and grayscale datasets is probably due to random statistical fluctuations caused by the limited number of runs.
   Overall, Dataset 3 turned out to be the best, therefore the High Pass FFT Gaussian filtering is the best pre-processing option among the ones described in Section 3.2;

4. Dataset 4 provided the worst outcomes, resulting in no improvements other than the ones under the external test. This strange behaviour appears to be associated with to the presence of elliptical galaxies in the used dataset.
   The Histogram Equalization and High-Pass Filter cause a huge distortion of the images, therefore many of the spiral, barred spiral or edge on galaxies could be turned into elliptically-shaped patterns. This observation can justify the outcomes of the three classes training: the network can't identify the different features of SA, SB and $EDGEON$ galaxies because the images are distorted, looking like elliptical patterns. However, the considered dataset doesn't feature elliptical galaxies, so the network assigns the same features to the three classes. Therefore, no real classification of the galaxies happens, but only a one out of three guess is made when considering both the internal and external validation set.
   In conclusion the results imply that the Histogram Equalization and High-Pass filtering do not improve the suitability of the dataset, but actually worsen it.

Chapter 5

Conclusions and Future Perspectives

In this thesis work, Deep Learning techniques were successfully implemented in order to automatically classify the morphology of elliptical, spiral, barred spirals and edge on galaxies.
This goal was achieved by using a Convolutional Neural Network built on a pre-existing MATLAB template, whose layer layout was modified. The image selection process has been described in detail and different pre-processing routines have been proposed and implemented, creating eight different datasets for the artificial neural network to use.
Amongst these pre-processing routines, the High-Pass FFT Gaussian filtering provided the best results, improving the suitability of the dataset to morphological analysis. Good results were also achieved by the High-Pass Gaussian Sharpening filter, that provided a slightly better suitability compared with the original dataset. On the other hand the combination of Histogram Equalization and High-Pass FFT Gaussian filtering provided the worst result, lowering the suitability of the dataset.
The results of the various training processes have been discussed, with particular emphasis on the diagnostic plots displayed. Overfitting behaviours have been identified and the results of this behaviur have been reported.
In the end, 50 runs of the training process were carried out, providing an outline on the efficiency of the various datasets, and making it possible to draw some interesting conclusions. The best results were provided by the High-Pass FFT filtered images, with a statistical accuracy of 43.68% for the classification of 4 classes of grayscale images and resulted in an accuracy of 42.64% when externally tested. These results correspond to a percentage variation improvement of 74.72% and 70.56% respectively, when referred to the random guess probability.
Despite not being completely satisfactory, the results obtained are very promising and provide proof that Deep Learning techniques could be the key to an automatic procedure for morphological classification of galaxies.

A possible extension of this work could imply an increase of the amount of images used both during training, validation and external testing processes. This improvement can be obviously achieved by selecting new images, but the consequences on training results of using mirrored or rotated versions of the already available images could be an interesting field of investigation too.
Furthermore, some improvements could be made on the pre-processing techniques: the implementation of routines different than the ones proposed in this thesis, could lead to better results of the training processes. The use of resampled images could also provide different outcomes.
Other interesting information about the behaviour of Deep Learning Neural Networks could be obtained by increasing the number of times the network is externally tested, perhaps providing different testing sets of images.
The last extension of this thesis work could be performing more runs of the training processes on the different datasets, in order to achieve more statistically significant results in the conclusions.

Acknowledgements

I wish to express my sincere thanks to my supervisor, Prof. Mauro Messerotti, for guiding me through this thesis work with patience and willingness. I would also like to thank all the COSMOS project group of INAF-Astronomical Observatory of Trieste, for providing me the MATLAB environment on the GPU accelerated computer and granting me the necessary run time.
In particular I would like to thank my supervisor, Dott. Daniele Tavagnacco, for his valuable help throughout all the problems I faced with the GPU accelerated computer.

Throughout all my life I've been extremely lucky, being surrounded by many kind, thoughtful and loving friends. These friends helped me through the toughest circumstances life has made me face, so I would like to heartfully thank them for being one of the most important parts of my life.
Regarding the last three years, I would like to thank all my friends in Trieste for supporting and helping me throughout this incredible journey.
My most sincere thanks go to Sara, Federico, Lorenzo, Riccardo, Stefano, Nicola and Albert: thank you for all the help and affection you all gave me to me throughout this adventure, I'm really grateful for having you in my life.
Some really special thanks go to my wonderful girlfriend Karin: thank you for being the most patient, kind and loving person I've ever had the fortune to meet, thank you for staying by my side throughout the last year and putting up with everything I do. Thank you for unconditionally supporting and encouraging me at all times. I love you.

Last but not least, I would like to thank my family for supporting me throughout all my life and the course of my university studies.
In particular, I would like to thank my brother Tommaso for inspiring me and pushing me to be the best version of myself I could possibly achieve.
My most special and heartfelt thanks go to my parents, Renato and Nicoletta, for all the love they gave me and for making me the person I am.
In particular, I would like to thank my mother for giving me the opportunity to study and for always providing me with everything I needed throughout all my life. Thank you for the unconditional support you gave me and for always believing in me. I couldn't be any luckier than having you as a mother. Words can't describe how much I love you.

Bibliography

[1] James Binney and Michael Merrifield. *Galactic Astronomy: James Binney and Michael Merrifield*. Princeton University Press, 1997.

[2] Hannu Karttunen, Pekka Kröger, Heikki Oja, Markku Poutanen, and Karl Johan Donner. *Fundamental astronomy*. Springer, 2016.

[3] Edwin Powell Hubble. *The realm of the nebulae*, volume 25. Yale University Press, 1982.

[4] SDSS. Sloan digital sky survey. `https://www.sdss.org/`. Online; Retrieved 2019-10-09.

[5] The de vaucouleurs system. `https://en.wikipedia.org/wiki/File:Hubble-Vaucouleurs.png`. Online; Retrieved 2019-10-09.

[6] Augustus Oemler. *The systematic properties of clusters of galaxies*. PhD thesis, California Institute of Technology, 1974.

[7] J Melnick and WLW Sargent. The radial distribution of morphological types of galaxies in x-ray clusters. *The Astrophysical Journal*, 215:401–407, 1977.

[8] Alan Dressler, Augustus Oemler Jr, Warrick J Couch, Ian Smail, Richard S Ellis, Amy Barger, Harvey Butcher, Bianca M Poggianti, and Ray M Sharples. Evolution since z= 0.5 of the morphology-density relation for clusters of galaxies. *The Astrophysical Journal*, 490(2):577, 1997.

[9] TC Beers and JL Tonry. Density cusps in clusters of galaxies. *The Astrophysical Journal*, 300:557–567, 1986.

[10] Michael R Merrifield and Stephen M Kent. The distribution of cluster members in the vicinity of a central dominant galaxy. *The Astronomical Journal*, 98:351–366, 1989.

[11] Michael Nielsen. Chapter 1: Using neural nets to recognize handwritten digits. `http://neuralnetworksanddeeplearning.com/chap1.html`. Online; Retrieved 2019-10-09.

[12] Mathworks. Introduction to machine learning, part 1: Machine learning fundamentals, . Online; Retrieved 2019-10-09.

[13] Mathworks. Introduction to machine learning, part 2: Unsupervised machine learning, . Online; Retrieved 2019-10-09.

[14] Mathworks. Introduction to machine learning, part 3: Supervised machine learning, . Online; Retrieved 2019-10-09.

[15] Carlotta Marchiori. An approach to pattern recognition in solar radio spectra by deep learning. University of Trieste, 2017-2018.

[16] Mathworks. Introduction to deep learning: What is deep learning? `https://www.mathworks.com/solutions/deep-learning.html`, . Online; Retrieved 2019-10-09.

[17] Mathworks. What is deep learning? `https://www.mathworks.com/discovery/deep-learning.html#whyitmatters`, . Online; Retrieved 2019-10-09.

[18] Mathworks. What is deep learning? `https://www.mathworks.com/solutions/deep-learning.html`, . Online; Retrieved 2019-10-09.

[19] Sambit Mahapatra. Towards data science: Why deep learning over traditional machine learning? Online; Retrieved 2019-10-09.

[20] Mathworks. Convolutional neural network. `https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html`, . Online; Retrieved 2019-10-09.

[21] Mathworks. Introduction to deep learning: Convolutional neural networks, . Online; Retrieved 2019-10-09.

[22] Mathworks. Create simple deep learning network for classification. `https://www.mathworks.com/help/deeplearning/examples/create-simple-deep-learning-network-for-classification.html`, Retrieved 2019. Online; Retrieved 2019-10-09.

[23] Mathworks. Deep learning toolbox. `https://www.mathworks.com/help/deeplearning/index.html?s_tid=CRUX_lftnav`, Retrieved 2019. Online; Retrieved 2019-10-09.

[24] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. Mnist database. `http://yann.lecun.com/exdb/mnist/`, Retrieved 2019. Online; Retrieved 2019-10-09.

[25] Wikipedia. File:mnistexamples.png. `https://commons.wikimedia.org/wiki/File:MnistExamples.png`, 2017. Online; Retrieved 2019-10-09.

[26] Nvidia tesla v100 tensor core gpu. `https://www.nvidia.com/en-us/data-center/tesla-v100/`. Online; Retrieved 2019-10-09.

[27] Application containerization. `https://searchitoperations.techtarget.com/definition/application-containerization-app-containerization`. Online; Retrieved 2019-10-09.

[28] Gpu vs cpu computing: What to choose? `https://medium.com/altumea/gpu-vs-cpu-computing-what-to-choose-a9788a2370c4`. Online; Retrieved 2019-10-09.

[29] Ned. `https://ned.ipac.caltech.edu/`. Online; Retrieved 2019-10-09.

[30] Strasbourg astronomical data center. `http://cdsweb.u-strasbg.fr/about`. Online; Retrieved 2019-10-09.

[31] Simbad astronomical database - cds. `http://simbad.u-strasbg.fr/simbad/`. Online; Retrieved 2019-10-09.

[32] Vizierr. `http://vizier.u-strasbg.fr/viz-bin/VizieR`. Online; Retrieved 2019-10-09.

[33] Aladin sky atlas. `http://aladin.u-strasbg.fr/aladin.gml`. Online; Retrieved 2019-10-09.

[34] NASA/IPAC. Digitized sky survey. `https://irsa.ipac.caltech.edu/data/DSS/`. Online; Retrieved 2019-10-09.

[35] J. E. Gunn, W. A. Siegmund, E. J. Mannery, R. E. Owen, C. L. Hull, R. F. Leger, L. N. Carey, G. R. Knapp, D. G. York, W. N. Boroski, S. M. Kent, R. H. Lupton, C. M. Rockosi, M. L. Evans, P. Waddell, J. E. Anderson, J. Annis, J. C. Barentine, L. M. Bartoszek, S. Bastian, S. B. Bracker, H. J. Brewington, C. I. Briegel, J. Brinkmann, Y. J. Brown, M. A. Carr, P. C. Czarapata, C. C. Drennan, T. Dombeck, G. R. Federwitz, B. A. Gillespie, C. Gonzales, S. U. Hansen, M. Harvanek, J. Hayes, W. Jordan, E. Kinney, M. Klaene, S. J. Kleinman, R. G. Kron, J. Kresinski, G. Lee, S. Limmongkol, C. W. Lindenmeyer, D. C. Long, C. L. Loomis, P. M. McGehee, P. M. Mantsch, E. H. Neilsen, Jr., R. M. Neswold, P. R. Newman, A. Nitta, J. Peoples, Jr., J. R. Pier, P. S. Prieto, A. Prosapio, C. Rivetta, D. P. Schneider, S. Snedden, and S.-i. Wang. The 2.5 m Telescope of the Sloan Digital Sky Survey. , 131: 2332–2359, April 2006. doi: 10.1086/500975.

[36] J. E. Gunn, M. Carr, C. Rockosi, M. Sekiguchi, K. Berry, B. Elms, E. de Haas, Ž. Ivezić, G. Knapp, R. Lupton, G. Pauls, R. Simcoe, R. Hirsch, D. Sanford, S. Wang, D. York, F. Harris, J. Annis, L. Bartozek, W. Boroski, J. Bakken, M. Haldeman, S. Kent, S. Holm, D. Holmgren, D. Petravick, A. Prosapio, R. Rechenmacher, M. Doi, M. Fukugita, K. Shimasaku, N. Okada, C. Hull, W. Siegmund, E. Mannery, M. Blouke, D. Heidtman, D. Schneider, R. Lucinio, and J. Brinkman. The Sloan Digital Sky Survey Photometric Camera. , 116: 3040–3081, December 1998. doi: 10.1086/300645.

[37] D. G. York, J. Adelman, J. E. Anderson, Jr., S. F. Anderson, J. Annis, N. A. Bahcall, J. A. Bakken, R. Barkhouser, S. Bastian, E. Berman, W. N. Boroski, S. Bracker, C. Briegel, J. W. Briggs, J. Brinkmann, R. Brunner, S. Burles, L. Carey, M. A. Carr, F. J. Castander, B. Chen, P. L. Colestock, A. J. Connolly, J. H. Crocker, I. Csabai, P. C. Czarapata, J. E. Davis, M. Doi, T. Dombeck, D. Eisenstein, N. Ellman, B. R. Elms, M. L. Evans, X. Fan, G. R. Federwitz, L. Fiscelli, S. Friedman, J. A. Frieman, M. Fukugita, B. Gillespie, J. E. Gunn, V. K. Gurbani, E. de Haas, M. Haldeman, F. H. Harris, J. Hayes, T. M. Heckman, G. S. Hennessy, R. B. Hindsley, S. Holm, D. J. Holmgren, C.-h. Huang, C. Hull, D. Husby, S.-I. Ichikawa, T. Ichikawa, Ž. Ivezić, S. Kent, R. S. J. Kim, E. Kinney, M. Klaene, A. N. Kleinman, S. Kleinman, G. R. Knapp, J. Korienek, R. G. Kron, P. Z. Kunszt, D. Q. Lamb, B. Lee, R. F. Leger, S. Limmongkol, C. Lindenmeyer, D. C. Long, C. Loomis, J. Loveday, R. Lucinio, R. H. Lupton, B. MacKinnon, E. J. Mannery, P. M. Mantsch, B. Margon, P. McGehee, T. A. McKay, A. Meiksin, A. Merelli, D. G. Monet, J. A. Munn, V. K. Narayanan, T. Nash, E. Neilsen, R. Neswold, H. J. Newberg, R. C. Nichol, T. Nicinski, M. Nonino, N. Okada, S. Okamura, J. P. Ostriker, R. Owen, A. G. Pauls, J. Peoples, R. L. Peterson, D. Petravick, J. R. Pier, A. Pope, R. Pordes, A. Prosapio, R. Rechenmacher, T. R. Quinn, G. T. Richards, M. W. Richmond, C. H. Rivetta, C. M. Rockosi, K. Ruthmansdorfer, D. Sandford, D. J. Schlegel, D. P. Schneider, M. Sekiguchi, G. Sergey, K. Shimasaku, W. A. Siegmund, S. Smee, J. A. Smith, S. Snedden, R. Stone, C. Stoughton, M. A. Strauss, C. Stubbs, M. SubbaRao, A. S. Szalay, I. Szapudi, G. P. Szokoly, A. R. Thakar, C. Tremonti, D. L. Tucker, A. Uomoto, D. Vanden Berk, M. S. Vogeley, P. Waddell, S.-i. Wang, M. Watanabe, D. H. Weinberg, B. Yanny, N. Yasuda, and SDSS Collaboration. The Sloan Digital Sky Survey: Technical Summary. , 120:1579–1587, September 2000. doi: 10.1086/301513.

[38] J. A. Frieman, B. Bassett, A. Becker, C. Choi, D. Cinabro, F. DeJongh, D. L. Depoy, B. Dilday, M. Doi, P. M. Garnavich, C. J. Hogan, J. Holtzman, M. Im, S. Jha, R. Kessler, K. Konishi, H. Lampeitl, J. Marriner, J. L. Marshall, D. McGinnis, G. Miknaitis, R. C. Nichol, J. L. Prieto, A. G. Riess, M. W. Richmond, R. Romani, M. Sako, D. P. Schneider, M. Smith, N. Takanashi, K. Tokita, K. van der Heyden, N. Yasuda, C. Zheng, J. Adelman-McCarthy, J. Annis, R. J. Assef, J. Barentine, R. Bender, R. D. Blandford, W. N. Boroski, M. Bremer, H. Brewington, C. A. Collins, A. Crotts, J. Dembicky, J. Eastman, A. Edge, E. Edmondson, E. Elson, M. E. Eyler, A. V. Filippenko, R. J. Foley, S. Frank, A. Goobar, T. Gueth, J. E. Gunn, M. Harvanek, U. Hopp, Y. Ihara, Ž. Ivezić, S. Kahn, J. Kaplan, S. Kent, W. Ketzeback, S. J. Kleinman, W. Kollatschny, R. G. Kron, J. Krzesiński, D. Lamenti, G. Leloudas, H. Lin, D. C. Long, J. Lucey, R. H. Lupton, E. Malanushenko, V. Malanushenko, R. J. McMillan, J. Mendez, C. W. Morgan, T. Morokuma, A. Nitta, L. Ost-

man, K. Pan, C. M. Rockosi, A. K. Romer, P. Ruiz-Lapuente, G. Saurage, K. Schlesinger, S. A. Snedden, J. Sollerman, C. Stoughton, M. Stritzinger, M. Subba Rao, D. Tucker, P. Vaisanen, L. C. Watson, S. Watters, J. C. Wheeler, B. Yanny, and D. York. The Sloan Digital Sky Survey-II Supernova Survey: Technical Summary. , 135:338–347, January 2008. doi: 10.1088/0004-6256/135/1/338.

[39] D. J. Eisenstein, D. H. Weinberg, E. Agol, H. Aihara, C. Allende Prieto, S. F. Anderson, J. A. Arns, É. Aubourg, S. Bailey, E. Balbinot, and et al. SDSS-III: Massive Spectroscopic Surveys of the Distant Universe, the Milky Way, and Extra-Solar Planetary Systems. , 142:72, September 2011. doi: 10.1088/0004-6256/142/3/72.

Appendix A

MATLAB Script of the Implemented Convolutional Neural Network

```
% all the images of the dataset are copied to another folder for safety
% reasons copyfile('/home/user/TESI/DATASET3/BASE/DATASET3-PASSA3/SA',
'/home/user/TESI/DATASET3/BASE/DATASET3/SAc')
copyfile('/home/user/TESI/DATASET3/BASE/DATASET3-PASSA3/SB',
'/home/user/TESI/DATASET3/BASE/DATASET3/SBc')
copyfile('/home/user/TESI/DATASET3/BASE/DATASET3-PASSA3/EDGE ON',
'/home/user/TESI/DATASET3/BASE/DATASET3/EDGE ONc')
copyfile('/home/user/TESI/DATASET3/BASE/DATASET3-PASSA3/ELLITTICHE',
'/home/user/TESI/DATASET3/BASE/DATASET3/ELLITTICHEc')

% 150 images are randomly selected and moved to another folder to be used
% by the training process

Dest = '/home/user/TESI/DATASET3/BASE/Temp2/SA';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SAc',
'*.png'));
Index = randperm(numel(FileList), 150);
for k = 1:150
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SAc',
FileList(Index(k)).name);
```

```
movefile(Source, Dest);
end

Dest = '/home/user/TESI/DATASET3/BASE/Temp2/SB';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SBc',
'*.png'));
Index = randperm(numel(FileList), 150);
for k = 1:150
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SBc',
FileList(Index(k)).name);
movefile(Source, Dest);
end

Dest = '/home/user/TESI/DATASET3/BASE/Temp2/EDGE ON';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/EDGE ONc',
'*.png'));
Index = randperm(numel(FileList), 150);
for k = 1:150
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/EDGE ONc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
Dest = '/home/user/TESI/DATASET3/BASE/Temp2/ELLITTICHE';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
```

```
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/ELLITTICHEc',
'*.png'));
Index = randperm(numel(FileList), 150);
for k = 1:150
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/ELLITTICHEc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
% End of the selection process
% The folder where the 600 images are is specified
DataSetPath = fullfile('/home/user/TESI/DATASET3/BASE/Temp2');
imds = imageDatastore(DataSetPath, ...
'IncludeSubfolders',true,'LabelSource','foldernames');

% the number of images in each subfolder is displayed
labelCount = countEachLabel(imds)
img = readimage(imds,1);
numTrainFiles = 125; %selection of the dimension of the training set
% each subfolder is splitted into training and validation sets
   imdsTrain,imdsValidation
= splitEachLabel(imds,numTrainFiles,'randomize');
% layer layout of the convolutional neural network
layers = [
imageInputLayer([550 649 1]);
convolution2dLayer(3,32,'Padding',1)
batchNormalizationLayer
reluLayer

maxPooling2dLayer(2,'Stride',2)
fullyConnectedLayer(4)
softmaxLayer
classificationLayer];

% training options for the network
options = trainingOptions('sgdm', ...
'ExecutionEnvironment','multi-gpu', ...
'MaxEpochs', 35, ...
'ValidationData',imdsValidation, ...
'ValidationFrequency',5, ...
'MiniBatchSize', 100,...
'Plots','training-progress', ...
```

```
'OutputFcn',@(info)savetrainingplot(info,i),...
'Verbose',false);
% network training
net1 = trainNetwork(imdsTrain,layers,options);
% accuracy computation
YPred = classify(net1,imdsValidation);
YValidation = imdsValidation.Labels;
accuracy1 = sum(YPred == YValidation)/numel(YValidation);
% random selection of the 25 external test images
Dest = '/home/user/TESI/DATASET3/BASE/Test2/SA';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SAc',
'*.png'));
Index = randperm(numel(FileList), 25);
for k = 1:25
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SAc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
Dest = '/home/user/TESI/DATASET3/BASE/Test2/SB';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SBc',
'*.png'));
Index = randperm(numel(FileList), 25);
for k = 1:25
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/SBc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
Dest = '/home/user/TESI/DATASET3/BASE/Test2/EDGE ON';
```

```
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/EDGE ONc',
'*.png'));
Index = randperm(numel(FileList), 25);
for k = 1:25
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/EDGE ONc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
Dest = '/home/user/TESI/DATASET3/BASE/Test2/ELLITTICHE';
filePattern = fullfile(Dest, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)
baseFileName = theFiles(k).name;
fullFileName = fullfile(Dest, baseFileName);
delete(fullFileName);
end
FileList = dir(fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/ELLITTICHEc',
'*.png'));
Index = randperm(numel(FileList), 25);
for k = 1:25
Source = fullfile('/home/user/TESI/DATASET3/BASE/DATASET3/ELLITTICHEc',
FileList(Index(k)).name);
movefile(Source, Dest);
end
% end of external test images selection
% the external set directory is specified
CheckSet = fullfile('/home/user/TESI/DATASET3/BASE/Test2');
testimg = imageDatastore(CheckSet, ...
'IncludeSubfolders',true,'LabelSource','foldernames');
% external test accuracy computation
YPredcheck1 = classify(net1,testimg);
YValidationcheck1 = testimg.Labels;

accuracycheck1 = sum(YPredcheck1 ==
YValidationcheck1)/numel(YValidationcheck1);
```

Appendix B

MATLAB Image Pre-Processing Routine

The following MATLAB script can provide all the image pre-processing routines described in this thesis work.

```
% Creation of the Gaussian High-Pass filter
% h1 = padarray(2,[2 2]) - fspecial('gaussian' ,[5 5],2);
% Specify the folder where the files live.
myFolder = '/Users/sebastianozagatti/Desktop/DATASET3/BASE/DATASET3';
% Check to make sure that folder actually exists.  Warn user if it doesn't.
if  isdir(myFolder)
errorMessage = sprintf('Error:  The following folder does not exist:%s',
myFolder);
uiwait(warndlg(errorMessage));
return;
end
filePattern = fullfile(myFolder, '*.png');
theFiles = dir(filePattern);
for k = 1 :  length(theFiles)

baseFileName = theFiles(k).name;
fullFileName = fullfile(myFolder, baseFileName);
imageArray = imread(fullFileName);
I = imread(fullFileName);
fullSourceFileName = fullfile(myFolder, baseFileName);

% Create destination filename
destinationFolder = '/Users/sebastianozagatti/Desktop/DATASET3/BASE/...
...  DATASET3-Filtered';
if  exist(destinationFolder, 'dir')
mkdir(destinationFolder);
```

```
end
for c = 1 :  3

% Grayscale Filtering
% J = rgb2gray(I);
% ---------------------
% High-Pass Gaussian Filter
% J = imfilter(I, h1);
% ---------------------
% histogram equalization
% II = histeq(I)
% ---------------------
% High-Pass FFT Gaussian Filter
% LL = fft2(double(I));
% L = fftshift(LL);
% for c = 1 :  3
% [M N]=size(LL(:, :, c));
% R=2; % filter size parameter
% X=0:N-1;
% Y=0:M-1;
% [X Y]=meshgrid(X,Y);
% Cx=0.5*N;
% Cy=0.5*M;
% Lo(:, :, c)=exp(-((X-Cx).2+(Y-Cy).2)./(2*R).2);
% Hi(:, :, c)=1-Lo(:, :, c); % High pass filter=1-low pass filter
% %Filtered image=ifft(filter response*fft(original image))
% end
% JJ=L.*Hi;
% K = ifftshift(JJ); % JJJ = ifft2(K);
% J = abs(JJJ);
% A = J(:,:,1); AA = J(:,:,2); AAA = J(:,:,3);
% imgNA = double(A-min(A(:)))  / double(max(A(:)-min(A(:))));
% imgNAA = double(AA-min(AA(:)))  / double(max(AA(:)-min(AA(:))));
% imgNAAA = double(AAA-min(AAA(:)))  / double(max(AAA(:)-min(AAA(:))));
% img(:,:,1) = imgNA; img(:,:,2) = imgNAA; img(:,:,3) = imgNAAA;
end
[sourceFolder, baseFileNameNoExtenstion, ext] =
= fileparts(fullSourceFileName);
outputBaseName = [baseFileNameNoExtenstion, '.png'];
fullDestinationFileName = fullfile(destinationFolder, outputBaseName);
% Write the image file.
imwrite(J, fullDestinationFileName) end
```